

**A Software Application for a Car
Sales and Servicing Company**

Mohammed Bham

BSc Computing (Industry)

2006/2007

The candidate confirms that the work submitted is their own and the appropriate credit has been given where reference has been made to the work of others.

I understand that failure to attribute material which is obtained from another source may be considered as plagiarism.

(Signature of student) _____

SUMMARY

The aim of this project was to develop a software application for Steve Graves Motors in order to solve the problems encountered in their current system of running and managing the business. Their current system consists of a paper based system whereby all their business processes and activities are carried out on paper, which are physically stored in their office. This causes many problems for the employees of the company, which are initially investigated in this project and are hoped to be solved.

The project involved following the prototyping methodology in order to develop the software application required. This consisted of gathering and analysing the user's requirements, and then through a number of iterations, designing, implementing, and testing the application with the users to identify whether their requirements have been satisfied. Finally the project involved evaluating the system to review the quality of the requirements that were implemented, in addition to the usability of the software according to Nielsen's five attributes of good usability.

The software application that was developed in this project was intended to solve the problems encountered in the current system of the company, by bringing together all their processes and activities so that they can be carried out in a single place, with all data stored and accessed centrally. In addition to this, a user manual was produced to assist the users' in using the various features of the software application. This report discusses the life cycle of the development of the software application that was carried out in this project.

ACKNOWLEDGEMENTS

I would like to thank my project supervisor Dr. Kristina Vuskovic for her guidance and support throughout the project, and my assessor Dr. Lydia Lau for her helpful feedback and suggestions in the mid-project report and the progress meeting.

I would also like to thank my parents and my family for their constant support and encouragement throughout all the years of my educational history.

My thanks also go to Steve Graves Motors for allowing me to do a project for them, and for their feedback and involvement in the project.

Finally I would like to thank all my friends who have supported me throughout the three challenging years at University, and for the countless number of memories and laughs we have had over the years.

CONTENTS

1. INTRODUCTION.....	1
1.1 PROJECT AIM	1
1.2 PROJECT OBJECTIVES	1
1.3 COMPANY BACKGROUND.....	1
1.4 PROBLEM DEFINITION	1
1.5 PROPOSED SOLUTION	2
1.6 MINIMUM REQUIREMENTS	3
1.7 FURTHER ENHANCEMENTS	3
1.8 DELIVERABLES	3
1.9 RELEVANCE TO DEGREE PROGRAMME.....	3
1.10 PROJECT SCHEDULE.....	4
1.11 STRUCTURE OF REPORT	4
2. BACKGROUND RESEARCH	5
2.1 EXISTING ALTERNATIVE SOLUTIONS.....	5
2.1.1 INTRODUCTION	5
2.1.2 AUTOVENDOR	5
2.1.3 DRAGON2000.....	5
2.1.4 RESOCO CARKEY	6
2.1.5 CONCLUSION	6
2.2 METHODOLOGY	7
2.2.1 INTRODUCTION.....	7
2.2.2 THE WATERFALL MODEL	7
2.2.3 THE SPIRAL MODEL	8
2.2.4 PROTOTYPING	8
2.2.5 RAPID APPLICATION DEVELOPMENT (RAD)	9
2.2.6 CHOSEN METHODOLOGY	10
2.3 USABILITY	11
2.4 DEVELOPMENT TOOLS	11
2.4.1 INTRODUCTION.....	11
2.4.2 PROGRAMMING LANGUAGE.....	12
2.4.3 DATABASE	13

2.4.4	CHOSEN DEVELOPMENT TOOLS	15
3.	ANALYSIS	16
3.1	FEASIBILITY ASSESSMENT	16
3.1.1	TECHNICAL FEASIBILITY	16
3.1.2	ECONOMIC FEASIBILITY	16
3.1.3	LEGAL FEASIBILITY	17
3.1.4	ORGANISATIONAL FEASIBILITY	17
3.2	USING ETHNOGRAPHY AND UML TO ASSESS THE CURRENT SYSTEM	17
3.2.1	IDENTIFYING THE USE-CASES	18
3.2.2	EXAMINING THE ACTIVITIES WITHIN THE USE-CASES.....	18
3.3	PROBLEMS WITH THE CURRENT SYSTEM	19
3.4	REQUIREMENTS GATHERING USING SQUIRO	20
3.5	REQUIREMENTS ANALYSIS USING MOSCOW RULES	23
4.	DESIGN & IMPLEMENTATION OF FIRST ITERATION.....	25
4.1	INTRODUCTION	25
4.2	DATABASE DESIGN	25
4.2.1	ENTITIES/TABLES	25
4.2.2	ENTITY-RELATIONSHIP DIAGRAM	26
4.2.3	DATABASE SCHEMA.....	26
4.2.4	NORMALISATION.....	26
4.3	GENERAL DESIGN ISSUES	27
4.3.1	SOFTWARE USABILITY	27
4.3.2	USER INTERFACE	28
4.3.3	NAVIGATION.....	28
4.3.4	COLOUR	29
4.4	FUNCTIONALITY DESIGN OF FIRST ITERATION	29
4.5	IMPLEMENTATION OF FIRST ITERATION	30
4.5.1	DATABASE IMPLEMENTATION	30
4.5.2	GENERAL SYSTEM IMPLEMENTATION	30
4.5.3	FUNCTIONALITY IMPLEMENTATION	33
4.6	TESTING OF FIRST ITERATION.....	34
4.6.1	UNIT TESTING	34
4.6.2	USER ACCEPTANCE TESTING	34
5.	DESIGN & IMPLEMENTATION OF SECOND ITERATION.....	35

5.1	INTRODUCTION	35
5.2	CHANGES MADE FROM FIRST ITERATION.....	35
5.3	DATABASE RE-DESIGN	36
5.4	FUNCTIONALITY DESIGN OF SECOND ITERATION	37
5.5	IMPLEMENTATION OF SECOND ITERATION.....	37
5.5.1	DATABASE ALTERATION	37
5.5.2	GENERAL SYSTEM IMPLEMENTATION	37
5.5.3	FUNCTIONALITY IMPLEMENTATION	38
5.6	TESTING OF SECOND ITERATION.....	40
5.6.1	UNIT TESTING	40
5.6.2	USER ACCEPTANCE TESTING	40
5.7	CHANGES MADE FROM USER TESTING.....	40
5.8	PRODUCTION OF USER MANUAL	41
6.	EVALUATION	42
6.1	INTRODUCTION	42
6.2	EVALUATION CRITERIA	42
6.3	EVALUATION OF USER REQUIREMENTS	42
6.4	USABILITY EVALUATION OF SYSTEM	45
6.5	COMPARISON WITH ALTERNATIVE SOLUTIONS.....	47
6.6	EVALUATION OF TECHNOLOGY USED.....	49
6.7	POSSIBLE FURTHER EXTENSIONS	49
7.	CONCLUSION	50
	REFERENCES	51
	APPENDIX A – PROJECT REFLECTION.....	54
	APPENDIX B – PROJECT SCHEDULE	56
	APPENDIX C – METHODOLOGY MODELS	58
	APPENDIX D – ANALYSIS	59
	APPENDIX E – DESIGN OF FIRST ITERATION	67
	APPENDIX F – TESTING OF FIRST ITERATION.....	73
	APPENDIX G – DESIGN OF SECOND ITERATION.....	78
	APPENDIX H – FINAL IMPLEMENTATION OF SYSTEM	81
	APPENDIX I – TESTING OF SECOND ITERATION.....	112
	APPENDIX J – EVALUATION	118
	APPENDIX K – USER MANUAL	122

1. INTRODUCTION

1.1 Project Aim

The overall aim of this project is to identify the problems encountered in the current system used by Steve Graves Motors for the running of their business, and then to analyse the requirements, design, and implement a software application to be used by the staff to assist them in the process of running and managing the company for their day to day activities. Therefore providing a possible solution to the problems encountered in the current system used by the company.

1.2 Project Objectives

In order to successfully satisfy the aims of this project, the following objectives must be completed:

- Identify the problems encountered in the current system in use of running the business.
- Research into a range of software development methodologies and choose to follow the most appropriate one relevant to this project.
- Research and evaluate the available software development technologies and databases.
- Gather and analyse the user requirements of the proposed new system.
- Design a software application incorporating the user requirements.
- Implement and develop a software application with a back-end database.
- Test the application at the implementation and user involvement stages of the project.
- Evaluate the new system to determine whether the user requirements have been satisfied.

1.3 Company Background

Steve Graves Motors specialises in the selling of a wide range of cars including small, medium, family, sports and 4x4 cars. Located in Birstall, West Yorkshire, the company's success and reputation is established by offering top quality cars at affordable prices to the general public. They also take bookings for cars and other vehicles for a repair or service to be carried out by a third party garage. The company currently comprises of three main staff; two of which are involved in all aspects of the everyday running of the business including sales, management, and administration; the third member is an accountant involved in the finance aspects of the company.

1.4 Problem Definition

The overall problem for Steve Graves Motors is that they have no centralised system for the running of their business. Most activities and processes of the business are paper based and independent of

each other. The only computer system currently used by the company is a website and a simple stock control system whereby they can add, edit and delete cars from their stock, in addition to a finance calculator for calculating the finance available to customers. These systems are not linked together.

The current process of selling a car requires the staff to write out all the details of the customer, car, and finance on a paper invoice, and then file these away. This makes searching for the details of a customer or a car sold several months ago a problem for the staff as they currently have to manually look through the filing cabinet and many invoices which can be a lengthy process. The company stores a range of cars offsite as well as onsite and therefore the customers that come to visit the showroom do not get to view the whole range of cars that Steve Graves Motors sell. This can deter away the customers as they may find that the company does not stock a car that satisfies their requirements, when in fact they possibly may do. In the current system the booking of a car for a repair or service is simply noted down on paper, this can easily be misplaced in the office within the various piles of paperwork. The VAT costs for each car sold by the company have to be individually calculated by the staff which in some cases can be prone to error as it is a lengthy repetitive process, and again results to more paperwork being produced that has to be filed away. These problems result in increased workload for the staff and the possibility of data being lost as it is physically stored in several places.

1.5 Proposed Solution

The proposed solution for solving the problems encountered by Steve Graves Motors is to develop a computerised system whereby most business processes can be carried out in a single place, and all data relevant to the company is stored centrally. This can lead to a reduction on the workload for staff and improve the efficiency of running and managing the business. It is hoped that the new system will be able to bring most aspects of the business together so that they can all be carried out together. Therefore be able to manage stock control for vehicles on a computer system, and then use this data to process the sale of a vehicle by adding the customer and finance details to an invoice, and finally storing these in a database where all the details of the sale can easily be retrieved in the future directly from the system. Additionally it is hoped that the new system could solve the other problems encountered in the current system including the ability to search for cars with a customer's specific requirements as the input, schedule appointments for a repair or service of a vehicle on the system, automatically calculate the VAT on cars sold and calculate the total amount of VAT owed, along with many other capabilities to improve on the current system. Therefore the proposed solution is hoped to eradicate the problems encountered in the current system and aid the staff in their daily activities and processes of running and managing the business.

1.6 Minimum Requirements

The minimum requirements of the project and the new proposed software application to be developed must have the ability to do the following:

- Manage vehicle stock control (add, edit, and view vehicles).
- Process, store, and invoice the details of the sale of a vehicle on a computer system (e.g. vehicle, customer, part-exchange, pricing).
- Schedule the appointments of vehicles for a repair or service.

Additionally the following must be included as another essential requirement of the project:

- Produce a non-technical user manual.

1.7 Further Enhancements

In order to further enhance the system to improve its capabilities and usability, the following functionalities could be added to the software application, if time permitting:

- Search current stock for a specific type of vehicle with the customer's requirements as input.
- Easily search for the previous sale of a vehicle and re-produce an invoice.
- Automatically calculate the VAT of each vehicle sold and calculate the total amount owed.
- Generate reports on sales performance for the business and compare with previous months.
- Invoice a repair or service with specific work carried out and parts fitted.
- Ability to add an image of a vehicle that has been added to the stock.
- View sales performance of vehicles and identify which are the most popular selling.
- Show how long a vehicle has been in stock and give warnings if unsold for a long time.
- Include a login and admin feature to access the software system by staff members only.
- Personalise the company's details on the system to suit the application to other companies.

1.8 Deliverables

The deliverables to be produced for this project are as follows:

- A software application.
- Software user manual.
- Project report.

1.9 Relevance to Degree Programme

There are many different skills that have been learnt throughout the degree programme that will have to be utilised to successfully complete this project. Learning about information systems concepts was

covered in the modules Introduction to Information Systems (IS11), and People Centred Information Systems (IS33). The knowledge and skills acquired in database technologies were covered in the modules Introduction to Databases (DB11), and Database Principles and Practice (DB21). The programming and software engineering skills were acquired in the modules Object-Oriented Programming (SE14), Object-Oriented Software Engineering (SE20), and Practical Software Development (SE24). As well as drawing upon the skills already learnt on the degree programme, there will also be many other new skills and knowledge that will be acquired as the project progresses.

1.10 Project Schedule

project schedule is an important aspect to a project like this as it helps the systems analyst and developer to better manage their progress and timing of the project, by breaking it down into a number of tasks that can be carried out at specific times up to the completion of the project. An initial schedule for this project was drawn up at the beginning of the project, however many changes were made to it during the course of conducting the project. This was mainly due to the decision to alter the methodology to follow for this project after the feedback gained from the mid-project report. The changes to the new schedule consisted of conducting the project in a number of iterations, whereby the design, implementation, and testing would be carried out for each group of requirements assigned to be developed into the system. Both the initial and revised schedules can be found in Appendix B.

1.11 Structure of Report

Various sections in this report refer to the appendix which gives more detailed information about what is being discussed in the report. This is because there are many parts of the project that had to be documented during the course of conducting this project, which could not all be included in the sections in the report. As a result, in many cases the report analyses and summarises the detailed information that it is referring to in the appendix. Therefore when the report mentions a reference to a section in the appendix, it is strongly advised to look at the appendix if further details and information are required to clarify what is being discussed in the report.

2. BACKGROUND RESEARCH

2.1 EXISTING ALTERNATIVE SOLUTIONS

2.1.1 Introduction

There are a range of software applications available to be used by car sales companies for the running of their business. A selection of these will now be reviewed to see whether any of them are suitable for use by Steve Graves Motors. The selection process of these systems consisted of searching the internet for software related to this project, and then choosing the three most relevant systems that could possibly be used by the company. An evaluation of these applications will also help in identifying whether any of their functionalities can be included in the software to be developed for this project.

2.1.2 AutoVendor

AutoVendor [9] is a low cost sales management system for small and independent car retailers. It has the ability to manage car sales on the computer system via a good graphical user interface and easy to use screens. The main features of AutoVendor include:

- Maintain a database on the stock of vehicles.
- Manage sales enquiries and part exchanges.
- Produce sales orders and invoices for customers.

Despite AutoVendor's good user interface, it is limited in the number of functions it can carry out. It does not provide the abilities to search a vehicle, search a sale, view sales performance or total VAT due, and schedule a vehicle for a repair or service. The cost of this system is £399 which is relatively cheap compared to its competitors. However as a result of the limited number of features of AutoVendor, it is not suitable for Steve Graves Motors to use.

2.1.3 Dragon2000

Dragon2000 [8] is an advanced software system and comprises of a range of modules that can be integrated together to make a complete system. Therefore a car sales module and a car workshop module can be integrated together to make a system that meets the requirements of a car sales and servicing company. The main features of Dragon2000 include:

- Sales and purchase invoices can be automatically printed.
- Ability to create profit and VAT reports.
- Handle part exchanges, and add parts and accessories to vehicles.
- Diary booking system for car repair or service.

- MOT and service next due reminders.
- All printed output is personalised to suit the company.

There are many benefits to using Dragon2000 because it includes many features that would satisfy the requirements of most car sales and servicing companies, including Steve Graves Motors. However a downside to this system is that it is relatively expensive. The integration of the car sales and workshop modules come to a cost of £1,390, with additional costs to be paid for support of the system at £29.50 per month, and training costs for the staff at £250 per person per day. Therefore Dragon2000 is not suitable for Steve Graves Motors as they do not have the budget to support such a huge investment identified in the feasibility assessment (section 3.2).

2.1.4 Resoco CarKey

The Resoco CarKey [7] used car sales software system is claimed to be simple to use, especially for staff who are not computer literate. CarKey provides quick and accurate pricing on used vehicles, and tools to make successful deals. The main features of CarKey include:

- Search for a particular type of vehicle.
- Access to vehicle history.
- Printing of details of base price, part-exchanges, deposits and balance.
- Management reports on stock valuation and sales performance.

Despite these features of CarKey, it does not include some of the functionalities required by Steve Graves Motors. These include the abilities not to handle the scheduling or invoicing of vehicles to be repaired or serviced, and calculating total VAT owed. In addition to this, the user interface does not seem very appealing from the screenshots on their website. Therefore Resoco CarKey is not a suitable system to use by Steve Graves Motors as it does not deliver all of the requirements of their business.

2.1.5 Conclusion

After evaluating the car sales software systems above, it is clear that none of them are suitable to be used by Steve Graves Motors. This is because they are either too expensive, or they do not meet all the requirements of the company, which was identified during the Analysis stage of this project (chapter 3). However an assessment of these has brought out a benefit to this project as it gives an idea to additional features that could be implemented into the application to be developed. This includes the ability to send service next due reminders to previous customers, and the ability to personalise the output on the printouts to suit the company's needs, which has been reflected as a possible further enhancement to this project in section 1.7.

2.2 METHODOLOGY

2.2.1 Introduction

A methodology is defined by Avison & Fitzgerald [1] as “*a collection of procedures, techniques, tools, and documentation aids which will help the systems developers in their efforts to implement a new information system*”; it is thought of as a set of processes to be followed in planning, analysing, designing, implementing, testing, and evaluating an information system. Many problems can occur in the development of information systems if a methodology is not followed. Avison & Fitzgerald [1] discuss that when methodologies were not used in the early years of software engineering, the needs of the users were not well established which led to unsuitable systems being developed for the users.

A methodology represents a way of developing information systems systematically and allows the systems analyst to accurately record the requirements of the user, as Avison & Fitzgerald [1] explain. Cadle & Yeates [2] suggest “*it is better to go into a project with a clear idea of the general form that the development is going to take*”. Therefore it is important to follow a methodology for this project as it will offer guidance in the whole process of building a software application for the user. A range of commonly used methodologies in information systems development will now be reviewed, and the most suitable one will be chosen to be followed for this project.

2.2.2 The Waterfall Model

The waterfall model is also known as the Systems Development Life Cycle (SDLC) and is considered to be the traditional way of developing information systems. The development of an information system is broken down into stages with each stage being completed before work starts on the next one, as described by Cadle & Yeates [2]. Therefore the outputs of one stage are used as the inputs for the next. The stages involved in a waterfall model are illustrated in the diagram in Appendix C.

There are strengths to following the waterfall model for the development of an information system. Sommerville [5] has identified that documentation being produced at the end of each stage is an advantage of the model, this is because it will help make sure that the requirements of that stage are complete and formally documented, which can then be presented to the developers for them to follow. Another advantage of the waterfall model is that the users and system analysts have the chance to review progress at the end of each stage, therefore if either feel that the requirements need to be changed then these can be discussed before proceeding to the next stage.

The waterfall model has come under criticism of having a reputation of moving from one stage to the next without looking back to see whether the previous stage was completed successfully. However Chester & Athwall [4] suggest regarding feedback in the model that *“iterations can take place within the traditional Waterfall life cycle”*, this means that systems analysts and developers can go back to the previous stages of the life cycle if they feel that they are inadequate and review this stage again. Another criticism of the waterfall model is that user involvement is only at the beginning and the end of the life cycle, this means that the user may not have any input while the system is in the process of being developed, and therefore important requirements not identified earlier may be left out.

2.2.3 The Spiral Model

The Spiral model was proposed by Boehm and *“introduces an evolutionary or iterative approach to the systems development”* when compared to the waterfall model as Cadle & Yeates [2] state. The requirements are gathered and the system is developed by performing the same activities over a number of cycles, whereas in the waterfall model the life cycle is all carried out once over a period of time. The Spiral is divided into four sections which can be seen in Appendix C. The process begins in the centre of the spiral and a phase of the development life cycle is represented within each loop as described by Sommerville [5], therefore the inner loop may identify system feasibility, the next loop may identify systems analysis, the next loop systems design, and so on up to the completion of the system. Each of the four phases in the model are also considered with the completion of each cycle.

The Spiral model *“introduces the important concepts of objective-setting, risk management and planning”* into the development life cycle as Cadle & Yeates [2] suggest. The factors that affect the outcome of an information system applies to all of these concepts which the spiral model identifies. This is therefore an important aspect for the management of the project as it could determine the success or failure of the development of an information system.

2.2.4 Prototyping

Chester & Athwall [4] state that a prototyping methodology *“produces a preliminary version of the required system that can be reviewed by end-users”*. This means that the system developers would build a prototype of the application based on the user’s initial requirements, and then present this to the users for them to test and review. When the user has reviewed the prototype system, they can suggest modifications and improvements to the system, which the developers can then go back and produce a better working system that meets the users’ requirements identified in the review. There are two methods of Prototyping, these are identified as:

- **Throw-away prototypes:** a prototype is developed for the users to review to identify their requirements, this is then discarded and the developers start to build a new system that incorporates the users' requirements, which will then lead towards the final system.
- **Evolutionary prototypes:** these are "*based on the idea of developing an initial implementation, exposing this to user comment and refining it through many versions until an adequate system has been developed*" as Sommerville [5] states. This means that an initial prototype is developed for the users to review, the user then makes suggestions for modifications or improvements, and these suggestions are then developed further into the prototype. This process is carried out until a final system is produced for the user.

The prototyping methodology provides benefits such as better user involvement during the development of the application, this means that they are also involved in helping to develop the system by suggesting areas of improvement as well as "*reveal errors and omissions in the requirements that have been proposed*" as they review the prototype, Sommerville [5] identified. Another benefit of Prototyping is that it presents the potential user interface of the application to the user at an early stage of the project. This allows the developers to modify the interface if it does not satisfy the users' needs before completing the final version of the system. Despite the benefits of prototyping, it also has a few disadvantages. The effort and time required in constantly developing prototypes can be wasteful as Avison & Fitzgerald [1] suggest. The large amounts of time spent in producing the interface and functionality of a prototype may be wasted if the user is not happy with it, therefore time will have to be spent again in re-developing the prototype. Hughes & Cotterell [3] state "*Lack of control*" as another drawback of Prototyping. This means that in the review, users can be more willing to suggest and try out new ideas which may be difficult to implement into the new system. This therefore takes away much of the control of the management of the project for the systems analysts and developers.

2.2.5 Rapid Application Development (RAD)

Rapid Application Development (RAD) is useful when a business needs "*to respond quickly to a changing and often uncertain environment*" as Curtis & Cobham [6] state. This means a business would require a system to be developed that can be used as soon as possible. A RAD approach offers this as it delivers a working prototype for them to use as soon as there is some functionality in the system, the working prototype is then improved and further developed when required. There are four phases to RAD as identified by Curtis & Cobham [6]:

- **Requirements Planning:** where the high-level management and strategic objectives are established, and where senior managers can make decisions on the strategy of the business.

- **Applications Development:** where users can take part in workshops and possibly by using a prototyping tool, the analysts can gather requirements which can be analysed and represented in data flow diagrams or entity modelling to assist in identifying user design.
- **Systems Construction:** where prototypes are produced and are assessed by the end-users in order to identify whether the interface and functionality of the system satisfy their requirements, if they do not then further modifications and improvements can be made.
- **Cutover:** where the users are given training, and the aim is to ensure the main functionality of the system is working. Any additional features can be developed at a later stage. The main functionality must be completed within an agreed period of time (timebox).

An advantage of using RAD is that it “*provides effective frameworks for taking and implementing difficult decisions*” as Curtis & Cobham [6] state, these may be decisions that have to be taken by senior managers if the requirements of the project are changed. Another advantage is that it provides a fast approach to the development life cycle, this can benefit a business especially when the requirements of the user are rapidly changing. A disadvantage to using RAD as Cadle & Yeates [2] suggest is that it can provide “*a lack of structure to the development*”, this means that the user requirements may not be defined clearly due of the fact that RAD is such a fast process. This may create difficulties for project managers as it may affect their control of the project.

2.2.6 Chosen Methodology

After analysing all the methodologies above, a conclusion can be made in regards to which methodology is best suited to be followed for this project. With the drawbacks of the waterfall model in that it does not provide much user involvement during the implementation and testing stages of the project, it is likely that the success of the project will be impacted due to the chance of the user’s requirements not fully being delivered or satisfied. The spiral model involves a lengthy process in following all the stages and procedures of the methodology in order to deliver an information system to the user, however due to the time constraints for this project, the spiral model would not be suitable. RAD emphasises mainly on projects that require a system to be implemented and delivered as soon as there is some functionality present in the prototype, however Steve Graves Motors do not require a part of the system to be delivered immediately.

Therefore the methodology that will be followed for this project will be evolutionary prototyping. This is because of the many benefits it provides, which includes the user involvement during the implementation and testing stages of the project. This will allow the system being developed to be

evaluated by the users in between various stages of the implementation before it is fully completed in order to identify whether their requirements are being satisfied. The prototyping methodology that will be followed in this project will first consist of gathering and analysing the user requirements, and then developing the system through a number of iterations whereby the design and implementation will be carried out for each group of requirements assigned for that iteration. Testing of these requirements implemented into the system will then be carried out at the end of each iteration, with relevant changes to be made at the next iteration. Finally the system will be evaluated to identify whether the user's requirements of the system have been successfully delivered and satisfied.

2.3 USABILITY

Usability is an important characteristic of a software application in addition to the functionalities of it. It involves the interaction a human has with the system via its interface, and the response from the system to the user. Nielsen [21] has stated five attributes to good usability which must be adhered to when developing the proposed new system, these are defined as the following:

- ***Learnability:*** in order for the user to begin to get some work done with the system quickly, the system should be easy to learn.
- ***Efficiency:*** once the user has learnt how to use the system, it should be efficient to use so that productivity from the system can be greatly increased.
- ***Memorability:*** the user must be able to use the system after a period of not using it, and easily remember how to use it again, therefore the system should be easy to remember.
- ***Errors:*** the system should prevent the chance of errors being made by the users when using the system, and be able to recover from any accidental errors made.
- ***Satisfaction:*** the users must be satisfied when using the system so that they like it and are happy with it, therefore the system should be pleasant to use.

2.4 DEVELOPMENT TOOLS

2.4.1 Introduction

The proposed system to be developed for Steve Graves Motors will consist of a front-end software application with a Graphical User Interface (GUI) where the user can directly interact with the system in order to carry out their necessary activities, and a back-end database which will be connected to the application to store all the data required by the business. The company currently has two desktop computers running Microsoft Windows, however the application and database will both be initially installed and run on a single computer at first, with the option of moving the database onto a dedicated

server and the front-end application installed on a number of machines in the future as the business further expands. The technologies required to develop these will now be analysed in order to choose which would be best suited for the requirements of the business, and the abilities of the developer.

2.4.2 Programming Language

In order to develop the front-end software application a programming language is required to code and create the software and the functionalities that it will be able to carry out. The chosen programming language must have the ability to create GUIs and to connect to a suitable database. Additionally, experience, familiarity and confidence with the programming language is equally important to ensure the software application is developed within the limited time available to meet the users' needs.

C#.NET

The C# programming language was introduced by Microsoft and was aimed to be integrated with the .NET Framework as Drayton et al [10] suggest. C# is an object-oriented programming language which evolved from C++ and has many similar features to the Java language. The aim for Microsoft was to make C# a simpler language than its competitors with fewer coding requirements. MSDN [11] state "*C# code is compiled as managed code*", this means it has improved security, better version control, and garbage collection. It has the ability to create GUIs using Microsoft Visual Studio, and the ability to connect to a database using an ODBC driver which is supported by the ADO.NET class provided in the .NET Framework, as explained by MSDN [12]. Developing software in C#.NET gives the application a Microsoft Windows look and feel, however due to the fact that it is part of Microsoft's .NET Framework, the applications developed in C#.NET are designed to run on Windows operating systems only. There is an open source project called Mono [23] which provides .NET applications to be run on other platforms, but it is not recommended to run .NET applications on a non-Microsoft operating platform due to compatibility and performance problems as Wise [24] suggests.

Visual Basic

Microsoft's Visual Basic programming language is now available as part of the .NET Framework for developing software applications. Developing applications in Visual Basic is particularly useful when a RAD methodology is followed as Roman et al [13] state. This is because Visual Basic allows developers to quickly build GUIs in Microsoft Visual Studio, and is relatively straightforward to use with much less difficult syntax than most other languages. This is backed up by MSDN [14] as they quote "*The Visual Basic language is designed to be human readable and accessible to everyone from novice programmers to advanced system architects*". Developers build software applications in Visual

Basic by drawing a GUI using the toolbox provided in Visual Studio, and then writing code to perform particular events that relate to actions performed from the GUI. It also has the ability to connect to a database using the ODBC driver. Visual Basic allows developers to build applications that are similar to the graphical nature of Microsoft Windows as Roman et al [13] suggest. However this also means that the applications are best run only on the Microsoft Windows operating systems as with C#.NET.

Java

Java is an object-oriented programming language developed by Sun Microsystems in the 1990's and its syntax is relatively similar to the C and C++ languages. The main advantage of Java is that it is platform independent, Wigglesworth & McMillan [15] state that Java allows developers to *“build programs that can run on a wide range of hardware architectures and host native operating systems”*. This means that software applications developed in Java can be run on any platform on any operating system, as long as the Java Runtime Environment is installed on the computer. Java offers a huge range of capabilities for developing software applications including the ability to create GUIs using the Swing framework components, and the ability to connect to databases using its JDBC driver. Choudhari [16] has identified a criticism of Java for being slow when compared to software applications developed in other programming languages. This is because the Java Virtual Machine must first interpret the binary code of Java into an instruction that the microprocessor can understand, this can take a short amount of time. However this should not be a major factor when considering a programming language as most computers these days are powerful enough to handle Java applications.

2.4.3 Database

A back-end database will be required as part of the new system in order to store the necessary data required by Steve Graves Motors. The database must have the ability to connect to the front-end software application so that data can be stored and retrieved by the user when they interact with the application. There are many important aspects to be considered before choosing a suitable database including cost, response speed, capacity, compatibility, reliability and performance.

Microsoft Access

Microsoft Access is one of the simplest and most flexible database management systems available to users. It is available as part of the Microsoft Office Professional software package and costs around £200 for a single license to be installed on a single computer. Access offers its own user-friendly graphical environment for the developer and user to interact with the database, as Atzeni et al [17] suggest. It is good for building simple applications that require small databases and is relatively easy

to use for novice developers due to it offering a GUI environment for development. However Access faces many problems when used for more complex software applications. Sinclair [18] explains that when large amounts of data are stored in Access, it can lead to slow data retrieval and poor performance as it requires more memory usage. Access is also not good for supporting multiple users', this is because when a user is updating a record, a lock is put on it so that other users cannot access the record at the same time. Additionally Sinclair [18] suggests that if Access crashes during a record transaction, then any changes made to it will be lost and will not be able to be recovered.

Microsoft SQL Server

Microsoft SQL Server is a relational database management system mainly used by medium to large businesses. The capabilities and performance of SQL Server are much greater than Microsoft Access, which is why it is preferred by most users. It provides more secure and reliable storage for both relational and structured data. However SQL Server is expensive and costs around £2,000 for the Standard edition, a free Express version is available for testing purposes but is not recommended for deployment to businesses. Advantages of SQL Server as identified by Sinclair [18] include the ability to support much larger databases with large amounts of data, handle lots of users accessing the same data without any major problems, and still have the ability to respond quickly to many other requests. This is mainly due to its client-server architecture. Sinclair [18] also states "*there is no conflict with different versions of the security implementation*", this is because all security is applied locally to the system, and all user and group information is held on the server in the master database.

MySQL

MySQL is the world's most popular open source database management system. It is free to download and is platform independent so it can be used on any operating system. The MySQL database system uses a client-server architecture that centres around the server. Butcher [19] states "*The server runs multithreaded*" and "*allows connections from multiple clients at once*". This means that the MySQL server program can be installed anywhere on one machine, and the MySQL client programs can be installed on a range of different machines to which they can connect to the server to store, manipulate, and retrieve data by sending SQL queries. A criticism of MySQL has been that it does not have many of the properties that can be found in other relational database systems. However the various advantages of MySQL as identified by DuBois [20] include speed with its fast response time, ease of use, capability of multithreading, portability as it is platform independent, size due to its large data capacity, cost of it being free, and support from many books and online user manuals available.

2.4.4 Chosen Development Tools

After analysing the various programming languages and databases available, it is clear that they all offer a wide variety of different features to suit the needs of many businesses. Despite the ability of Microsoft Access to offer a quick and simple desktop application system with a database and a GUI for the user, it is not suitable to meet the requirements of the proposed solution. This is simply because Steve Graves Motors is an expanding business and requires large amounts of data to be stored, as a result the performance of Access will suffer if data needs to be constantly stored and retrieved. Access is best used as a single desktop application, however as the business expands the proposed application will require to be installed on a number of machines for the staff to use with the database on a separate server, Access will not be efficient at this point due to its performance and costing issues.

The C# and Visual Basic programming languages have the ability to produce a front-end software application that meets the requirements of Steve Graves Motors, however as I do not have any experience in using these languages, it is not appropriate to choose one of these to develop the application as it would take time to learn and adapt to them within the limited time available to produce the application. The Java programming language has the ability to allow a software application to be developed that meets all the requirements of the business, and as I already have experience and a good knowledge in using Java, it is decided that Java will be used to develop the front-end software application for Steve Graves Motors. Despite Microsoft SQL Server being a good database management system, it is simply not feasible to use this as the back-end database for the company because it is too expensive for them. It is therefore decided that MySQL will be used as the database for the proposed system because it is free and offers many of the capabilities and performance required by the business. As Java and MySQL are also platform-independent, they will still be compatible to easily run on a different operating system if the business decides to change to a free open-source operating system if they purchase new computers in the future.

An Integrated Development Environment (IDE) is a type of software that provides developers with a platform to write, edit, and test source code when building software applications. Eclipse offers many features expected from a leading IDE including a syntax highlighter, incremental code compilation, debugger, class navigator, project manager, and source control systems, as identified by Storkel [22]. As the developer in this project already has experience in using Eclipse, it will therefore be used as the IDE to develop the front-end software application with Java for this project.

3. ANALYSIS

3.1 Feasibility Assessment

A feasibility assessment is a study of *“whether the proposed system will be cost-effective from a business point of view and whether it can be developed within existing budgetary constraints”*, as Sommerville [5] suggests. This means that a study can be carried out analysing whether the new system will realistically be affordable, beneficial, and feasible to put into practice by the business. After assessing these aspects for Steve Graves Motors, a decision can be made on whether to continue with further progression of the proposed system and project.

3.1.1 Technical Feasibility

Avison & Fitzgerald [1] discuss that the proposed system must be able to function with the current technology available in the market and that it can be built with sufficient expertise. As discussed in section 2.3 of this report, it was concluded that Java and MySQL were going to be used to develop the system for Steve Graves Motors. These technologies are technically feasible as they both have sufficient capabilities to carry out a number of functions required by the business. Additionally as I have experience in using these technologies, I therefore have enough expertise to build a system that meets the users’ needs. Steve Graves Motors already have desktop computers that are capable of running the proposed software application, therefore the current hardware technology in place is sufficient. Avgerou & Cornford [25] state *“the selection of technology has to be carefully undertaken to be certain that the organization can master and sustain it”*, this means that the technology to be chosen for the proposed system must have the ability to be easily used by the employees of the business. The employees of Steve Graves Motors only need to know how to use the front-end software application that will be developed in Java as this is the only interaction that will be required by them. This should not be a problem as Java allows good graphical user interfaces to be developed.

3.1.2 Economic Feasibility

An economic feasibility assessment determines *“whether a proposed information system is financially affordable and if it is going to lead to economic benefits”*, as Avgerou & Cornford [25] suggest. This means that the application to be developed for the business must be affordable by them, and the benefits of using the system in the company must outweigh the costs associated with purchasing and running it. Kendall & Kendall [26] discuss that if the long-term benefits of a system do not overshadow the short-term or running costs, then the project should not proceed any further as it is not

economically feasible. Steve Graves Motors do not currently have a budget to afford a new IT system, therefore the existing alternative systems currently available are not suitable. The tools required to develop the software application for Steve Graves Motors will not cost them anything as they are all open source, and therefore freely available. Additionally they will not incur any costs in the actual development of the application as this is being done for them for free as part of this project.

3.1.3 Legal Feasibility

Assessing the legal feasibility for a project involves determining whether the new system will infringe any national or international company laws as Avison & Fitzgerald [1] suggest. This means that the new system must oblige to the requirements and conditions that have been legally imposed and not break any of these laws. There are no legal issues for the proposed system for Steve Graves Motors as the technologies used for the system can be obtained legally for free as they are open source available. Additionally the new system will not breach the Data Protection Act as information held on the system would not be shared with anyone outside the organisation. Therefore the project is legally feasible.

3.1.4 Organisational Feasibility

Organisational feasibility refers to the *“changes that a proposed information system entails for the structure of an organisation”*, as Avgerou & Cornford [25] state. This is essentially an assessment of how much the new system will change the structure of an organisation in terms of its staff and the processes required to carry out the everyday business activities. The proposed software application for Steve Graves Motors will not affect the employees or the structure of the organisation in having to make any changes. The company comprises of three employees and implementing the application into the company will not require any reductions in staff or additional staff to be hired, the current employees can use the new system to carry out their activities as they would have normally done with a paper based system. The project is therefore organisationally feasible.

3.2 Using Ethnography and UML to Assess the Current System

An important reason as to why it is important to assess the current system used by Steve Graves Motors is that it can help identify the problems associated with it, which can then be used to make sure that they are avoided when implementing the proposed new system. The combination of using ethnography and UML will help identify the problems associated with the current system. An assessment of Steve Graves Motors's main activities and processes are discussed in this section, and the problems identified and associated with them are discussed in section 3.4.

Ethnography is “*an observational technique that can be used to understand social and organisational requirements*”, as Sommerville [5] states. It requires an analyst to observe and involve themselves in some of the employee’s daily activities in an organisation, this is so that the analyst can get a first hand experience of how staff normally conduct their work. By doing this the analyst becomes part of the situation, and only then can they truly understand and interpret what is happening, as Bennett et al [27] discuss. Therefore for this project, ethnography will be used to identify what the processes are and who is involved in the current system for Steve Graves Motors by observing and participating in the working environment, then presenting the findings in the form of a number of UML diagrams to understand the current system more clearly. UML (Unified Modelling Language) is used to represent business situations and systems in the form of diagrams to give an overview of the subject being discussed. By using UML to represent the business situations and the current system for Steve Graves Motors, it will help to better understand what the problems are, where they occur, and why they occur.

3.2.1 Identifying the Use-Cases

Before assessing the details of the current system, it is important to identify the main activities of the current system, and who its users are. This is so that the scope of the system can be documented, and from which the details of each activity can be further investigated. By observing what the employees of Steve Graves Motors do along with asking them questions, the users of the system and the activities they carry out were identified. A Use Case Diagram was then drawn up in UML to present the scope of the current system by showing the users and their activities within their working environment, as shown in Appendix D. The stick figures are called *actors* which represent the users of the system, and the ellipses which are called *use-cases* represent the activities/functions of the system. The lines from the users to the use-cases represent that the user interacts with a particular function of the system. From the use case diagram representing the working environment for Steve Graves Motors, we can see that an employee of the company carries out all of the activities/functions mentioned in the current system. By identifying these activities/functions (use cases) in the Use Case diagram, the use cases can now be further examined in detail to identify exactly how they are carried out.

3.2.2 Examining the Activities within the Use-Cases

The activities that are carried out within each use case can be represented in UML diagrams in the form of an Activity Diagram. As Bennett et al [27] suggest, activity diagrams “*consist of a set of actions linked together by flows from one activity to the next*”. A rectangle with rounded edges represents an activity, a diamond represents a decision point, and an arrow represents the flow of

activities and decisions. In order to adopt the ethnography technique effectively for this project, I asked one of the employees if I could be involved in a few of the activities (use-cases) of the business, in order to get a first hand experience in identifying how a particular process is carried out and what problems are encountered with it. I was given the chance of performing the activities for the Add Vehicle use-case because it is not as complex and does not require important customer interactions. The activities for the Sell Vehicle use-case were identified during one of the visits to the company by observing a sale being processed by an employee of Steve Graves Motors when a customer was purchasing a vehicle. The activities involved in the other four use-cases were identified using a combination of observing and involvement by myself. A description of all these use-cases, and activity diagrams for the Add Vehicle and Sell Vehicle use-cases can all be found in Appendix D.

3.3 Problems with the Current System

After assessing the processes and activities of each of the use cases for Steve Graves Motors using the ethnography technique and activity diagrams in UML, the main problems associated with them can now be outlined as follows:

- **Data Replication:** the same data is replicated across many areas of the company. When a new vehicle is added to the stock, its details need to be stored in the supplier invoice, stock book, the computer stock system, and its sale invoice when it is sold. For a sale, the customer's details are stored in the sale invoice and in the stock book. Financial figures are replicated across invoices and the stock book. This can lead to data retrieval problems as data may need to be combined from different places to get the required information.
- **Data Inconsistency:** as data is stored across many areas of the company, if data in one place is changed and the same data is not updated in other places it is stored at, then that data will be inconsistent and incorrect in those places. This can be a real problem when updating profit/loss figures as they may lead towards inaccurate targets for future sales.
- **Increased Workload:** from the activity diagrams for Add Vehicle and Sell Vehicle, it is clear that the employees of Steve Graves Motors have a lot of work to carry out for simple jobs. An example is in the Sell Vehicle use case when a customer is buying a car on monthly finance, the employee is required to fill in all the customer's details again on the finance form, additionally if there is a part exchange, the employee is again required to go through the whole process of adding all the details of the Add Vehicle use case.
- **Increased Effort:** for the simple tasks of the business, the employees have to spend large amounts of time and effort to carry them out. Searching for a previous sale should be a simple task, however the staff are required to go through the whole stock book manually, looking

through hundreds of sale entries for a particular sale they are looking for. Additionally an employee is also required to fill in two sale invoices when selling a car.

- **Increased Paperwork:** from the number of invoices that have to be written and stored when adding or selling a vehicle, it results in a huge amount of paperwork that has to be filed away. This can later lead to physical storage problems and some paperwork being misplaced or lost. Also, as scheduling a car for a repair is just noted down on paper, this too can easily be misplaced or lost as it is not formally stored in a particular place.

These problems with the current system that have been identified need to be avoided when developing the new system for Steve Graves Motors. This will be critical when coming to evaluate the success of the system and the project as these problems encountered in the current system must be eliminated.

3.4 Requirements Gathering using SQIRO

The requirements of the users for new system to be built for Steve Graves Motors were gathered using a range of techniques which are often referred to as a tool called SQIRO. These are common techniques often used to gather the user requirements of a system to be built. The choice of techniques to be used from SQIRO depends on the company the system is being built for, as some of the techniques may not be suitable or relevant for a particular company or project.

Sampling

Sampling refers to the method of collecting blank or completed documents from the company the system is being built for, and determining what the inputs and outputs are of the various processes carried out within the business from the documents collected, as Bennett et al [27] suggest. This information can then be used to help in assessing what inputs and outputs are required for the new system. During one of the visits to Steve Graves Motors, a blank vehicle sale paper invoice which is usually used to write and store the details of the sale of a vehicle was collected, this can be found in Appendix D. The vehicle sale invoice will help in identifying what data and information will be required to input and store into the new system when processing a sale. From the invoice we can see that for a customer buying a vehicle, their name, address, and contact phone number is required; for the vehicle being sold or a vehicle being part-exchanged, its make, type, chassis number, registration number, date it was first registered, colour, and speedometer reading are required; for the pricing of the vehicle, its price, net allowance, deposit, and amount due from customer are required. Additionally the buyer and seller (company employee) are required to sign and date the invoice. The new system must therefore allow the data that is usually entered in these fields on the paper invoice to be inputted and stored into the new system in order to process the sale of a vehicle.

Questionnaires

Gathering data from questionnaires usually involves system analysts to present the users with a set of written questions to which the users mark their responses. A criticism of questionnaires as Maciaszek [28] states is that “*questionnaires are less productive than interviews*”, this is because the user filling out the questionnaire cannot clarify their answer further than what the question asks. Questionnaires are useful for collecting quantitative data in large projects that involve a lot of potential users, because it would be hard to organise interviews with many users. As Steve Graves Motors only consists of three employees, questionnaires were not used to gather the user requirements for this project.

Interviews

Interviewing users for gathering their requirements can be of two types as Sommerville [5] explains, they can be closed interviews where the analyst proposes a set of predefined questions to the user who responds with their answers, or they can be open interviews where the analyst explores a range of issues with the user who gives their own opinions and thoughts to what they require from the system. An advantage of interviews is that “*personal contact allows the analyst to be responsive and adapt to what the user says*”, as Bennett et al [27] state. An interview was conducted with both employees of Steve Graves Motors together in order to identify their requirements of the new system. The interview that was conducted was a combination of a closed interview and an open interview. The closed interview involved making sure that they were asked what functionalities they would require in the new system that is presently carried out in the current system, what additional functionality they require that is not normally carried out in the current system, and how they would like the interface and usability of the system to be. The open interview involved them being able to further develop their answers to get a more detailed knowledge on exactly what they require, this was done by proposing new questions that relate to their previous responses. The set of questions and summarised answers from the interview can be found in Appendix D.

The interview with the two employees of Steve Graves Motors identified many functionalities that they require from the new system, what data/information and processes are required in each functionality, and how the usability of the system should be. These requirements identified in the interview can all be found in the summary of the employees’ responses in Appendix D, which are also defined in section 3.6 of this report.

Reading

Background reading within requirements gathering involves the analyst reading up various documents within an organisation to give them a better understanding of how it operates. This can give some information about the current system from which requirements of the new system can be identified. Bennett et al [27] state that “*written documents do not often match up to reality*”, this means that when a company document states how a process should be done, in practice it is usually not carried out in that way by the employees. This could be because the document may be out of date or employees may prefer a different way to carry out tasks which they find easier. It was decided that the background reading technique was not suitable for requirements gathering in this project as there are not any formal process documents held by Steve Graves Motors.

Observation

Using observation for gathering user requirements is an effective technique in circumstances “*where the business analyst finds it difficult to obtain complete information through interviews and questionnaires*”, as Maciaszek [28] suggests. This is because a user may find it difficult to communicate their requirements without actually showing the analyst their concern. The observation technique was carried out for this project in order to assess the current system and the problems associated with it (as discussed in sections 3.3 and 3.4), and to try and find any other user requirements that were not identified when using the sampling and interviewing techniques. Observation proved to be useful in this project when identifying desirable features that could be added to the new system by observing what functionality the current computer stock system offers. During the visit to the company when the interview was carried out, one of the employees showed how the current computer stock system and said that a good feature of it is that it shows how many days a vehicle has been in stock for and not yet sold. In the vehicle stock table showing a list of all the vehicles in stock, if a vehicle has been in stock for less than 30 days then a box next to the vehicle entry in the table is highlighted in green, if it has been in stock between 30-90 days then the box is highlighted in yellow, and if a vehicle has been in stock for over 90 days then the box is highlighted in red. This gives the employees an indication of how long the vehicle has been in stock for and that something should be done if the vehicle is going into the yellow or red categories, an option would be to reduce the price of the vehicle. This is a very useful feature of the current computer stock system and therefore would be desirable if it could be implemented into the new software application.

3.5 Requirements Analysis using MoSCoW rules

Using the MoSCoW rules for analysing the requirements of the new system “*ensures that a critical examination is made of requirements and that no ‘wish lists’ are made by users*”, as stated by Avison & Fitzgerald [1]. The MoSCoW rules stand for Must Haves (Mo), Should Haves (S), Could Haves (Co), and Wont Haves (W) of a particular system to be developed. The purpose of the rules is that they prioritise the requirements into a set of categories in terms of which are the most and least important. This can be useful for deciding which requirements of the system the developer should concentrate on more and complete before certain other requirements. The findings from the assessment of the user requirements gathered for the new system to be developed for Steve Graves Motors using the SQIRO techniques can be categorised into these rules, which will outline what functionalities of the system are more important than others and which should be prioritised when developing the software application.

Must Haves

Must Haves are the requirements of a system that must be implemented when the system is complete, and if they are not included then the system will not function as Bennett et al [27] suggest. For the application to be developed for Steve Graves Motors, it was identified from the interview that the system must include the following required features:

- Manage vehicle stock control having the ability to add, edit, and view vehicles.
- Process, store and invoice the details of the sale of a vehicle on the computer system.
- Schedule the repair or service of a vehicle with information stored on system.
- Include a user manual to aid in using the system.

These are essentially the minimum requirements of the project and must be satisfied for the project to be of any success. These will therefore be completed before starting work on the other requirements.

Should Haves

Should Haves are the requirements that will be of benefit if they are delivered, but the success of the project does not rely on them, as Avison & Fitzgerald [1] suggest. The requirements that should be included in the proposed application for Steve Graves Motors include:

- Ability to search for a particular type of vehicle with a number of specific vehicle specifications entered into the system.
- Ability to search for a previous sale with the registration number of the vehicle sold entered into the system.
- Automatically calculate VAT of each vehicle sold and total VAT due every few months.

- Automatically calculate the profit or loss made at the end of each month and allow a comparison to be made with previous months.
- Make the system secure so that the software application can only be accessed via a login feature by the staff members.

Could Haves

Could Haves are requirements that “*are desirable but provide less benefit to the user*” as Bennett et al suggest [27]. This means that it would be good if these requirements are delivered, but the success of the project will not be affected much if they are not delivered. The requirements that could be included in the new system for Steve Graves Motors (if time permitting) that were identified in the interviewing and observation techniques include:

- Ability to invoice a vehicle that has been repaired or serviced with specific work carried out, specific parts fitted, and the total price broken down into relevant sections.
- When adding a vehicle to the stock, allow an image of that vehicle to be added to the system.
- For the vehicle stock control system to show how long a vehicle has been in stock and give warning signs if they have been in stock and unsold for a large amount of time.
- Ability to view and compare the sales performance of different types of vehicles by their make or body style in order to see which are the most popular and best selling vehicles.

Wont Haves

Wont Haves are requirements that can be left out and not be implemented in the initial delivery of the system, however they can possibly (but not necessarily) be developed and added to the system at a later date, as Avison & Fitzgerald [1] suggest. The feature that was decided not to be implemented in this delivery of system for Steve Graves Motors includes the ability to give an indication of when a customer’s vehicle’s service is next due after they have scheduled their vehicle for a previous service. Although the idea for this feature was derived from the Dragon2000 software (identified in section 2.1.3), this was not seen as an important requirement by the staff of Steve Graves Motors. Additionally with the short amount of time available to build the system for the company, there will probably not be enough time to implement this as the other more important requirements will take priority.

4. DESIGN & IMPLEMENTATION OF FIRST ITERATION

4.1 Introduction

As the methodology followed in this project was evolutionary prototyping, the development of the software application for Steve Grave Motors was completed in a number of iterations. Each iteration consisted of designing, implementing, and testing with the users the functionalities of the system assigned for that iteration. This chapter discusses the first iteration that was carried out which includes the *Must Haves* of the system as mentioned in section 3.6 of the report (the minimum requirements), apart from producing the user manual which was done after the system was fully developed.

4.2 Database Design

As identified earlier, a database will be needed as part of the new system to store all the data required by Steve Graves Motors. A good database design is crucial to a successful information system as it will ensure that the objectives of the project and the requirements of the new system are met. The key design issues of the required new database to be created will now be discussed.

4.2.1 Entities/Tables

After the users' requirements were defined in the analysis section of the report, the tables (entities) required in the new database can now be identified. The tables required in the database were identified as follows for the first iteration of the prototyping methodology being followed:

- **Vehicle:** to hold the details of a vehicle.
- **Pricing:** to hold the pricing details of a vehicle.
- **Supplier_Customer:** to hold details of a supplier or buyer of a vehicle.
- **Appointment:** to hold the details of a vehicle scheduled for a repair or service.

It was initially thought that separate supplier and customer tables would be required, however it has been decided that a suppliers details and a customers details are best put in the same table for a number of reasons. Firstly a suppliers and customers details to be stored will be of very similar format, i.e. name, address, phone, etc. Secondly, and more importantly, when a customer buys a vehicle and part-exchanges their old vehicle, the supplier of the part-exchanged vehicle is effectively that customer. Therefore their details will only need to be stored once as a customer and a supplier. This will be a huge advantage because if their details need to be changed in the future, then it will only need to be changed in one place, rather than having to update their details in two places if they were stored in separate customer and supplier tables. The decision to have a separate Pricing table to store the pricing

details for a vehicle was simply because there are many pricing attributes required to be stored, and therefore it would be much more sensible to keep all pricing details in a separate table for neatness, rather than cluttering the Vehicle table with too many attributes that could lead to possible confusion.

4.2.2 Entity-Relationship Diagram

An entity-relationship diagram is a graphical representation of a database system which shows the entities within the database and the relationships that exist between them. The final E-R Diagram that was drawn for the database required in this project (both first and second iterations) can be found in Appendix G along with a commentary of it explaining the relationships that exist between the entities.

4.2.3 Database Schema

A database schema defines the attributes that exist in the tables of a database and therefore what type of data is stored. It also shows what will be the primary keys and foreign keys in the tables. The schema for the database of the new system for Steve Graves Motors required in the first iteration is outlined below, the primary keys are underlined and the foreign keys are in italics.

- Vehicle (reg, make, model, version, engine, colour, doors, body, fuel, transmission, mileage, date_reg, chassis, date_bought, date_sold, sold, *supplier_id*)
- Pricing (reg, method, company, loan_amount, duration, monthly, price_bought, selling_price, price_sold, exchange_price, amount_to_pay, amount_paid, amount_due)
- Supplier_Customer (supp_cust_id, name, address1, address2, address3, postcode, phone, email, save_supp, *reg_bought*, *reg_exchanged*)
- Appointment (name, phone, reg, make, model, engine, work_type, date, work_required)

When a vehicle is to be added to the stock, the relevant attributes in the Vehicle and Pricing tables will be populated with the data entered for that vehicle. In the Supplier_Customer table, if a new supplier is added, the relevant attributes will be filled in except for the *reg_bought* and *reg_exchanged* attributes. Additionally when a vehicle is sold, the *date_sold* and *sold* attributes in the Vehicle table for that vehicle will be populated, and the customer's details will be entered into the Supplier_Customer table along with the registration numbers of the vehicle they have bought and/or part-exchanged in the *reg_bought* and *reg_exchanged* attributes.

4.2.4 Normalisation

Poorly constructed databases are prone to a number of problems, most notably functional dependencies and anomalies. If the same data is stored in more than one place, it would cause an update or a delete anomaly whereby if data is updated or deleted in one place, then it will also have to

be updated or deleted in another place. Normalisation provides a solution to this problem by having tables in a state called Third Normal Form (3NF). In Third Normal Form, data about a particular item is only required to be stored in one place, with its primary key determining all its other attributes. This therefore eliminates the possibility of update and delete anomalies, and data would only need to be changed in one place. The database to be built for the new system is designed in such a way that data about a particular item will only need to be stored in one place and access to it is required by its primary key only. Additional tables to store a vehicle's registration and a supplier or customer id is not required because there is only ever one distinct vehicle, which is supplied by one supplier only, and bought by one customer only. Therefore by storing just the supplier id in the vehicle table and the vehicle registration in the Supplier_Customer table with a customer's details, this is sufficient enough to ensure that data is kept independent, consistent, reliable, and not unnecessarily replicated.

4.3 General Design Issues

4.3.1 Software Usability

As well as designing the functionality of the new system for Steve Graves Motors, it is also important to design the usability of the system as discussed in section 2.3 of the Background Research chapter. Nielsen's [21] five attributes of usability will be used as a guide to design the usability of the software:

- **Learnability:** Clearly state how functions can be carried out by giving simple, easy to understand labels and instructions to buttons and fields.
- **Efficiency:** Ensure that the system allows tasks to be carried out in similar ways so that they can be carried out quicker once users have learnt how to use the software. Minimise the amount of data entry required with fields already filled in where necessary.
- **Memorability:** Keep the interface and layout consistent by keeping fields, tables, and buttons in their same respective places. Allow similar functions to be carried out and navigated in the same way as other functions.
- **Errors:** Provide confirmation boxes to users when carrying out tasks such as deleting vehicles or sales so that they can confirm what they are doing is correct and not accidental, also provide information boxes when users try to perform an illegal operation.
- **Satisfaction:** The system must allow the tasks carried out by the staff to be easier than conducted with the current system. Users must be satisfied with the system in how it enables them to perform tasks, this can be reviewed during the system evaluation.

All these usability attributes will be taken into consideration and implemented when developing the software. Their effectiveness will be evaluated at the end of the project.

4.3.2 User Interface

Designing the graphical user interface of the front end software application for Steve Graves Motors is equally important as any other design issue. This is because the only interaction the staff of the company will have with the system will be via its user interface. Maciaszek [28] emphasises that consistency within interface design is highly essential, he mentions that users “*should be presented with a familiar and predictable environment*” and that the positioning of components should be similar throughout the different screens in the application. The software application to be developed in this project will be designed so that components such as buttons will always be in the same place throughout the different screens. Minimal user input is also an important design issue when developing software applications because it can reduce user errors and improve the speed of data entry as Bennett et al [27] suggest. They state that this could be assisted by the use of lists rather than re-entering values again, the application will therefore be designed to use listing components such as drop down combo boxes where necessary rather than input text fields. Bennett et al [27] also mention that user input could be minimised by avoiding having to make the user to input data where it could be obtained automatically, this design feature will be implemented into the system in places such as where the date field can be automatically filled in with the date derived from the system. Another important design issue suggested by Bennett et al [27] is appropriate user support, they mention that the system interface should provide assistance when the user has made an error or does not know what action to take. This can be done by providing help or error messages to guide the user in the correct action to take. The system will therefore be designed so that it provides assistance to the users such as when deleting an item, a confirmation box is provided, or when some required data entry fields are left blank, then an information message will be provided telling them some required fields are empty.

4.3.3 Navigation

A good design of the navigation of a software application plays a key role in the usability of the system. Maciaszek [28] states that “*the user should never feel lost among opened windows*”, therefore it is important that the software application for Steve Graves Motors is designed in a way that the user will know what screen they are on. This can be achieved by having clearly stated titles at the top of each screen indicating what screen the user is on and what action can be performed at it. The buttons must be labelled in simple to understand text in where it will take the user to, additionally a ‘Back’ button must also be included on every screen which will enable the user to go back to the previous screen they were on. Good system navigation is also achieved by a good design of its menus and toolbars, as Maciaszek [28] suggests. In addition to having a main menu screen, the system will be designed so that its main functionality screens can be accessed from any place the user is at in the

application. This will be achieved by including a menu bar at the top of each screen where the user can select where they would like to go. It is also important that the user should not have to navigate to their desired screen via many other screens, the number of mouse clicks to navigate around a software application must be as minimal as possible. The software application will therefore be designed so that the user can access all parts of the system in a maximum of three mouse clicks, which is suitable.

4.3.4 Colour

The appropriate use of colour on a software application gives it an attractive look and invites user friendliness. Kendall & Kendall [26] discuss the importance of having a good contrast between foreground and background colours. Their findings show that good practice is to have four to seven different colours, and to have darker colours for text on a lighter background. For the system in this project, it is decided that five colours will be used in a consistent manner. They are light blue, light soft yellow, and light brown for various backgrounds, with black and dark blue for text foregrounds.

4.4 Functionality Design of First Iteration

This section of the report discusses the design considerations that were made before implementing each functionality assigned for the first iteration of the prototyping methodology followed.

Vehicle Stock Control

There will be a number of screens that will make up the management of the vehicle stock control functionality of the system. Each screen will have different buttons to navigate the user to different parts of the system. The main screen for the stock control functionality will consist of a table showing all vehicles currently in stock. From here the user will be able to go to the screens for adding or viewing a vehicle which will consist of a number of tabs separating the data for a vehicle or supplier. The edit vehicle screen will be similar to the add vehicle screen but with the relevant fields filled in with the vehicles details, and it will be accessible when the user is at the view vehicle screen. A detailed design of these functionalities can be found in Appendix E.

Process Sale of Vehicle

The screen for selling a vehicle will also contain a number of tabs including Vehicle, Customer, Part-Exchange, and Pricing to separate the different types of data entry required. When the button for processing the sale will be clicked, the user will be informed that a sales invoice has been written to a HTML file from where they can print it off to give to the customer as a receipt. Additionally the

details of the sale will be able to be retrieved on the sales history screen which will contain a table of all the previous sales made, and from where the user will have access to view its full details. A detailed design of these functionalities can again be found in Appendix E.

Schedule Vehicle for Repair or Service

The main screen for scheduling a vehicle for a repair or service will consist of a table showing all the future appointments, a feature to check the availability of a selected date, and buttons to add or view an appointment. The screen for adding a new appointment will consist of two tabs, Customer & Vehicle, and Work Required. A detailed design of this can be found in Appendix E.

4.5 Implementation of First Iteration

This section discusses the implementation of the system in the first iteration. The design issues mentioned earlier were taken into consideration during the development of the database and the software. Screenshots of the system were only taken after the system was implemented in the second iteration after the extra functionalities were included, but they can still be found in Appendix H.

4.5.1 Database Implementation

The tables in the database schema that was designed in this first iteration were created in MySQL. In order to create the tables a number of *create* statement queries in SQL had to be written in the MySQL command line client. This creates the tables and the columns within it with their respective data storage types. Where normal text needed to be stored, the column had a *varchar* data type as this allows alphanumeric data to be stored. All the fields that store prices were given the data type *double*, as this allows decimal point figures to be stored which would be required if a price contains pence as well as pounds. The *supp_cust_id* column in the *Supplier_Customer* table was set to *auto-increment* because as each supplier or customer is added in this table, the column will automatically increment by 1 and each record will have different values which will be uniquely identifiable. All the primary key fields were set to *not null* (in addition to some other fields) so that a record cannot be stored without its primary key being populated for identification.

4.5.2 General System Implementation

This section discusses the general implementation issues of the software system that had to be implemented consistently throughout various parts of the system, which provide the baseline and foundations to implement the required functionalities of the software application. The implementation of the specific functionalities required in the system are described in section 4.5.3. As decided in the

Background Research chapter, the Eclipse IDE was used to write the code to develop the application in with the Java programming language. Every detail about the implementation of the software could not be included in these sections, therefore only the main implementation issues are discussed.

Database Connectivity via JDBC Driver

In order for the front end software application to retrieve and insert data from the database, a connection has to be established. The JDBC driver allows the connection from a Java application to any type of database. Within the Eclipse IDE, the driver had to be added as an external jar file. In addition to this, a text file called *jdbc.properties* had to be included, which holds information about the database location, name, username, and password to gain access. These can be simply accessed from the *DBConnection* class that was created which includes a method to establish a connection and allow querying the database. Once a connection is established, queries can be made to the database. The method to create a query consists of the following code:

```
connection = getConnection();
Statement statement = connection.createStatement();
ResultSet results = null;
try {
    results = statement.executeQuery(query);
} catch (java.sql.SQLException erg) {
}
return results;
```

Within the code, 'query' is a string which is taken in as a parameter of the method. When this method is accessed from any other class to query the database, the query to be submitted is passed as a parameter, the method executes the query, and the results are returned in a ResultSet.

Graphical User Interface

The interface of the system was created by using a *JFrame* component which is part of Java's Swing framework. This creates a blank screen where panels can be added to it that contain other GUI components (such as text fields, tables, tabs). Once the frame has been created, panels containing other components can be added to it. Panels are created using a *JPanel*, and they have to be added to a Container's content pane, which in turn can then be added to the frame. The following code was used to add panels to a frame:

```
frame.getContentPane().removeAll();
frame.getContentPane().add(panel);
frame.setVisible(true);
```

This code is contained within a method. This is because when switching to other screens, the screens will contain different panels to each other. Therefore when a different screen needs to be displayed,

this method is called which removes the panel of the previous screen, and then adds the new panel of the screen required to the frame. The panel to be added is passed as a parameter to the method.

Adding Components

Java's Swing toolkit provides the creation of many GUI components which can be added to the screen. The main component required to navigate around the system is a button. In Java a button is created using a *JButton* component. To set an action on it whereby when it is click a method is called to perform a function or action, an *ActionListener* has to be added to it. The code that was used to create a button and to set an action upon it consists of the following:

```
JButton button = new JButton("Button Label");
button.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent event)
    {
        // call method to perform action
    }
});
```

As discussed in the design section, tabs are required to separate the data presented on the screen. Each tab consists of a panel added to it which contains a number of other components such as labels and text fields. The tabs were created using a *JTabbedPane* component in Java. Many different individual tabs can be added to the main tabbed pane. The code to create the tabs for the add vehicle screen consisted of the following format:

```
JTabbedPane addVehicleTabs = new JTabbedPane();
addVehicleTabs.addTab("Vehicle", vehicleComponents());
addVehicleTabs.addTab("Supplier", supplierComponents());
```

In the code above, once the *addVehicleTabs* tabbed pane is created, the Vehicle and Supplier tabs are added to it. This is done by calling the *addTab* method on the pane, and passing parameters into it containing the tab's label, and the panel to be added to it. Some of the screens on the system contain tables. Tables were created by using the *JTable* component in Java. In addition to a table, a *JScrollPane* had to be created to which the table had to be added. This is because a scroll pane acts as a 'wrapper' around the table so that if the number of rows in the table exceeded the table size, the scroll pane allows the user to scroll down the table. The code to create the vehicle stock table and scroll pane consisted of the following:

```
JTable stockTable = new JTable(allRows, columnNames);
stockTable.setRowHeight(30);
stockTable.setFont(new Font("Dialog", Font.PLAIN, 18));
stockTable.setAutoResizeMode(JTable.AUTO_RESIZE_SUBSEQUENT_COLUMNS);
stockTable.getTableHeader().setReorderingAllowed(false);
JScrollPane scrollPane = new JScrollPane(stockTable);
```

In the above code when the table is created, two Vectors are passed as parameters. The first contains each vehicle's details retrieved from the database in a number of elements which are represented as rows in the table, and the second contains the column names. The row height was set to be larger at a height of 30, and the font size of the text was also set to be larger at 18.

4.5.3 Functionality Implementation

The implementation of the required functionalities of the system for Steve Graves Motors are briefly discussed in this section. These include the functionalities of the system that were designed in section 4.4 (the minimum requirements of the system). The details of how these were implemented could not be included in this section due to the amount of detail required to describe the implementation of these features, however a detailed description of the implementation of these with screenshots and code snippets can be found in Appendix H, pages 84 - 99. A detailed explanation on how these functions work and how they are used within the system can be found in the User Manual in Appendix K.

Manage Vehicle Stock Control

The functionality of the system to manage vehicle stock control was implemented first. A screen was created which included creating the first set of tabs. Text fields and combo boxes were also created on this screen to allow the input of data. Next the main stock control screen was implemented which includes a table to show all the vehicles in stock after they have been added. Finally the abilities to view and edit vehicles selected from the stock table were implemented.

Process Sale of Vehicle

The ability to process the sale of a vehicle was then implemented. Tabs were again created which contained text fields to allow input for the information required to process a sale. A class which writes the details of the sale to a HTML file was then created, this is so that when the Process Sale button is clicked, an invoice is written which can be printed out. The screen that contains the table of all previous sales was created, along with the abilities to view and edit a previous sale.

Schedule Vehicle for Repair or Service

The final main functionality of the system which includes the ability to schedule a vehicle for a repair or service was then implemented. A screen was created which included tabs and text fields to allow the input of data required. Next the main screen of this functionality which includes a table showing all the current appointments was implemented. A function to check the availability of a date before adding a new appointment was also created, this consisted of creating a number of combo boxes and a

button where a specific date can be selected and checked to view all the appointments on that date. Finally the ability to view and edit an appointment was implemented.

4.6 Testing of First Iteration

Testing is an important aspect of software development. It ensures that errors made by the programmer are identified and corrected as early as possible, therefore reducing the chance of further errors being made. The testing of the implemented system carried out during the first iteration is now discussed.

4.6.1 Unit Testing

Unit testing is when “*individual components are tested to ensure that they operate correctly*”, as Sommerville [5] states. This means that as each component of the system is developed, it is tested to see whether it carries out what was expected from it correctly or not. During the development of the software for Steve Graves Motors, unit testing was constantly carried out as each component was developed. For each test, a note was made on what was being tested, what was expected from it, and what its result was. If the result was not as expected, the reason for its failure was noted and was then corrected immediately until that component was working correctly. A list of the main components of the system that were tested along with their results can be found in Appendix F.

4.6.2 User Acceptance Testing

The purpose of carrying out user acceptance testing is one of the most important aspects when following the prototyping methodology. This is because after the implementation of the functionalities in each iteration, they can be tested by the user to see whether they satisfy their requirements correctly or not before proceeding to further developing the system. If the functionalities implemented so far do not meet their needs, then they can be re-implemented at the next iteration. Sommerville [5] too suggests that acceptance testing can identify any problems of the system that do not satisfy the user's requirements. After the implementation of the system's functionalities was carried out in this first iteration, it was taken to the employees of Steve Graves Motors for them to test. They provided feedback on certain aspects of the system which they thought required modifying or improving, these were all noted and a summary of them can be found in Appendix F. These feedback suggestions were then intended to be implemented during the next iteration of the project's methodology. Apart from the suggestions they made, they accepted that the other functionalities satisfied their requirements.

5. DESIGN & IMPLEMENTATION OF SECOND ITERATION

5.1 Introduction

This chapter discusses the second iteration that was carried for the prototyping methodology followed in this project. It firstly consists of the changes that were made to the system from the feedback gained from the user testing carried out in the first iteration. It then discusses the design and implementation of the *Should Haves* and *Could Haves* of the new system as mentioned in section 3.6 of the report (the further enhancements). The testing that was carried out is then summarised, and the implementation changes made from the feedback gained from the user testing are finally discussed.

5.2 Changes Made From First Iteration

The improvements and modifications suggested in the user acceptance testing carried out in the first iteration were implemented first, shortly followed by the additional features. The text field used for inputting data into the Transmission field was changed to a drop down combo box, and the content of the box was populated to include options for selecting Manual or Automatic. Next a new column named 'Colour' was added to the stock list table, and the query to retrieve the data about a vehicle was modified to include its colour. The field that displayed the price bought at the view vehicle details screen was removed, however at the edit vehicle screen this field was kept as customers are not shown this screen. The function to add a customer of a vehicle as a potential future supplier included adding a checkbox in the customer tab for when processing the sale of a vehicle. When this is ticked the save_supp field is set to 'yes' for the customer's record in the Supplier_Customer table. For checking whether the amount to pay, amount paid, and amount due fields are not left blank when processing the sale of a vehicle, an *if* statement was used in the code to check that if any of them are left blank, then display an error message and do not process the sale. Finally, to give an indication as to exactly which item is being viewed or edited on the screen, the titles at the top of the screen were set to include the variable that stores the vehicle's registration number, so that it can also be displayed in the title.

For implementing the additional features suggested from the feedback gained in the user testing, the ability to edit a supplier was implemented first. For this extra feature, a button on the Add Vehicle screen was created which when clicked, a small window is opened where the user can select from a drop down list which supplier they wish to edit. When a supplier is chosen, their current details are displayed in the relevant text fields, from where the user can change their details and then click the Update button which will update the supplier's details in the database. For the ability to re-create an invoice for a vehicle already sold, a button on the view sale details screen was created which when

clicked, creates a new HTML file with the current details about the sale. The same code for when creating a sales invoice when a vehicle sale is processed was used for efficiency, however a method was used specifically for this function in that the original declaration for a customer's signature is removed as this is not needed just for a re-print of the invoice. The final additional feature requested from the user that was implemented at this stage of the project was the ability to print out the details of a vehicle currently for sale. A button labelled Print Vehicle Details was included on the view vehicle details screen which when pressed, a HTML file is created that writes all the details of the vehicle from the database to the file. This can then be printed off as normal.

5.3 Database Re-Design

For some of the functionalities of the system to be implemented in the second iteration, the database that was originally built has to be re-designed in some areas to accommodate these functionalities. The additional tables (entities) that will be required are as follows:

- **Stats:** to hold the details of the sales performance and popularity of vehicles.
- **Login:** to hold the username and password for login access to the system.
- **Company:** to hold details of the company such as name, address and contact details.

An entity-relationship diagram of the final design of the database required for the new system for Steve Graves Motors can be found in Appendix G. The schema for these new tables are:

- Stats (item, type, total_stock, sold, remaining, total_costs, total_income, percentage)
- Login (username, password)
- Company (company_name, address1, address2, phone, fax, max_appointments)

In the Stats table, the *item* attribute will hold values such as a vehicle's make or its body type, and the *type* attribute will hold a value on whether that item is in fact a make or body type. In addition to adding these new tables to the database, a few of the current tables also have to be slightly altered to include a few more attributes. Firstly the Vehicle table was re-designed to include the attribute *notes* (to store any additional information about a vehicle), and the attribute *image* (to store a file-path name to an image of the vehicle added to stock). Additionally in the Pricing table the attributes *profit* and *vat* were added to the design to store the profit and VAT of a particular vehicle when sold. These will be required for when implementing the functionality for the system to automatically calculate the total VAT owed, and for when comparing profit/loss figures. The Appointment table will have to be altered to include a number or *work* and *price* attributes, this will be required for when implementing the functionality to invoice a vehicle with specific work carried out and their prices. In addition to this, a total price and VAT field will need to be added.

5.4 Functionality Design of Second Iteration

Before the required functionalities were implemented into the software application in the second iteration, they had to be designed first. As before, the general design guidelines in section 4.3 of the report were taken into consideration when designing these. Due to the vast amount of functionalities that were required to be implemented, the details of their design could not all be included in this section. However a summary of their design issues that was carried out can be found in Appendix G.

5.5 Implementation of Second Iteration

This section discusses the implementation of the system that was carried out in the second iteration. Screenshots of the system were taken during this iteration, and they include the functionalities of the system developed in the first and second iterations. These can be found throughout Appendix H with code snippets and a detailed commentary on how they were developed.

5.5.1 Database Alteration

Before creating the new tables required for the second iteration, the current tables were altered to the specified design in section 5.3. To add the extra fields required in these tables, a number of *alter* statements in SQL had to be written in the MySQL command line client. This would alter the table and would add these extra fields along with their data types. Next, the three new tables as required from the design in section 5.3 were created using a number of *create* statements in SQL. A screenshot of all the tables created in MySQL can be found in Appendix H.

5.5.2 General System Implementation

The second iteration required the implementation a number of general additions that are consistent throughout the system such as colour, images, navigation and better presentation of the interface. These are discussed in this section with screenshots of their outcome available in Appendix H.

Adding Colour

Adding colour to areas of the system required setting the background colour of a component. Setting the background colour of the window to a light blue was done by specifying the RGB colour value for the main panel on every screen. Adding a soft light yellow colour to the tabs required setting the background colour of the panel to be added to that tab to its relevant RGB value. A light brown colour for the tables required setting its background colour to an RGB value. The text that displays information in the screens for viewing the details of a vehicle, sale, or appointment was set to a dark

blue, this was done by setting the foreground colour of the label that contains the text. The code for setting the colours to these components are of the following format:

```
mainPanel.setBackground(new Color(216, 228, 248));
tabPanel.setBackground(new Color(255, 255, 225));
table.setBackground(new Color(236, 233, 216));
label.setForeground(new Color(49, 106, 197));
```

Adding Images

Random images of close-ups on cars were included throughout the system. To implement these on the screen, an *ImageIcon* object had to be created for each image which is referenced by the image's file path, and then the object had to be added to a normal *JLabel* component which in turn is added to the screen. The code to include images on a screen is as follows:

```
ImageIcon vehicleImage = new ImageIcon(filePath + "fileName.jpg");
JLabel imageLabel = new JLabel(vehicleImage);
```

As well as adding images of vehicle close-ups to the screen, icons were added to the tabs to allow easier differentiation. This required the creation of the tabs from before to be modified. An *ImageIcon* object referenced by the icon's file path had to be created as mentioned above, and then this icon is passed as a parameter to the *addTab* method. The code consists of the following:

```
addVehicleTabs.addTab("Vehicle", vehicleIcon, vehicleComponents());
```

Menu Bar

A menu bar was implemented as part of the system which is included at the top of every screen. This allows the user the ability to navigate to any other main part of the system from whichever screen they are currently on. A snippet of the code to create the navigation to the stock control screen consists of the following, some parts of this code was used to create the navigation to the other screens:

```
JMenuBar menuBar = new JMenuBar();
JMenu navigate = new JMenu("Navigate");
menuBar.add(navigate);
vehicleStock stock = new vehicleStock();
navigate.add(stock);
frame.setJMenuBar(menuBar);
```

5.5.3 Functionality Implementation

The implementation of the functionalities of the system assigned for the second iteration were carried out according to the design for each of them mentioned in section 5.4, and the general design issues discussed in section 4.3. At the end of the second iteration, the functionalities of the system assigned

for the second iteration were implemented. These are the *Should Haves* and *Could Haves* of the system as mentioned in section 3.6 of the report, which are essentially the further enhancements of the project. Due to the vast amount of functionalities that were required to be implemented in this iteration, the details of how they were developed could only be briefly discussed in this section of the report. However as mentioned before, a commentary with code snippets and screenshots of the implementation of these, along with the printouts that the system produces for some of the functionalities can all be found in Appendix H, pages 100 - 111. The specific details of how they work can be found in the User Manual in Appendix K. The functionalities developed were:

- ***Search Vehicle:*** this is available by clicking the Search Vehicle button on the main stock control screen, which brings up a new window whereby the user can select a number of requirements from the options in the combo boxes. When Search is click, the results that match the selections are displayed in the stock table.
- ***Search Sale:*** when the Search Sale button on the previous sales screen is clicked, a small input box appears, and when the registration of the vehicle sold is entered and OK is clicked on the box, the sale is displayed in the table.
- ***Calculate VAT:*** this can be carried out by selecting a range of dates from a box on the VAT Calculation screen, which will then display all the details of the vehicles sold in that period in the VAT table. The total VAT due is displayed in a small table at the bottom of the screen. The ability to write the details to a HTML file to be printed off was also implemented.
- ***View Sales Performance:*** by going to the Sales Performance screen, the user can select a range of dates whose sales figures in that period are be displayed in the table. When the user selects another range of dates, the figures for that period are also added in the table.
- ***Invoice Vehicle Scheduled:*** after a vehicle has been scheduled and its details are to be edited, a Work Performed and Pricing tab are added where details about work carried out and prices can be entered. An invoice can then be written to a HTML which can be printed off.
- ***Login Access:*** each time the software is run, a login box appears first where the user is required to enter the username and password. If correct, the main menu screen appears. The password can also be changed at the Administration screen of the system.
- ***View Sales Performance of Vehicles:*** this feature is available by going to the Vehicle Sales Performance screen where all statistics on vehicles sold are displayed. A combo box was included on the screen to allow the user to change the view from vehicle make to body style.
- ***Add Image of Vehicle:*** when adding or editing a vehicle, a button was included in the Image tab so that when clicked by the user, a small window appears where they can select the image file to add along with the details of the vehicle. This image is then displayed in the Image tab.

- ***Personalise Company Details:*** at the Administration screen, text fields were included that are filled in with the company's current details. The user can change these and click the Update Company Details button at the bottom of the screen which will update the details.

5.6 Testing of Second Iteration

This section discusses the testing that was carried out after the system was implemented in the second iteration. The testing in this iteration was similar to the testing that was carried out in the first iteration. This consisted of unit testing of the system, and user acceptance testing.

5.6.1 Unit Testing

As mentioned before, unit testing involves testing each individual component of the system as it is being developed to see if it is functioning correctly. While the functionalities of the system assigned for the second iteration were being implemented, unit testing of these was constantly carried out. The component being tested along with its result was noted for each test. A summary of all the unit tests that were carried out during the second iteration can be found in Appendix I.

5.6.2 User Acceptance Testing

User acceptance testing was again carried out after the functionalities assigned for the second iteration were implemented. As this was the last iteration where the required functionalities of the system were to be implemented, it was vital that the employees of Steve Graves Motors tested the system so that any final changes could be made before it is fully finished, and handed over to the business for them to evaluate. Full details of the feedback gained from the user testing can be found in Appendix I. The main improvements to the system the users would like are to include separate total price fields when invoicing a vehicle that has been scheduled for work to be carried out, including commas in the price figures on the search vehicle screen, and the ability to change the type of contact outputted on the printouts. The employees of the company also specified that they would like a feature to re-select the dates when calculating the total VAT due, the ability to print out sales performance of the business, and the ability to change the username set on the system. The users accepted the fact that the other functionalities satisfied the requirements.

5.7 Changes Made From User Testing

After the user acceptance testing was carried out in the second iteration, the required changes to the system were implemented. For creating separate total price fields when invoicing a vehicle scheduled for work to be carried out, additional text fields were created in the Pricing tab. The system was also

implemented to calculate the separate prices and fill in the figures in the relevant fields when the Calculate Total Price button is clicked. Commas in the figures in the mileage and price combo boxes on the search vehicle screen were included to easily differentiate the figures in thousands. The price figures stored in the database do not include commas, therefore a *replace* method had to be called on the string for the price selected in order to replace the comma with an empty character, this would then allow a compatible comparison with the price in the database. For the ability to change a contact type, the company table in the database had to be slightly modified. The fax column was replaced with *contact2_value*, and an additional *contact2_type* column was added. Additionally an extra text field had to be included on the administration page to allow the contact type to be changed, which would be displayed on future printouts. The additional feature required of including a button on the VAT calculation screen to display the dates selection box simply involved the task of creating a new button on the screen, and then adding an *ActionListener* method to it whereby when it is clicked, the box to change the dates is displayed. To include the feature of printing out a sales performance report required creating a new method where all the details from the table are written to a HTML file to be printed off. A button was also created which calls the method to write the details to the file. The ability to change the username for the system required a new text field to be created on the administration page. When the option to change the username is selected, and a new username is entered in the field, the username will be changed as long as the old username and password entered are correct. All these changes made to the system are reflected in the screenshots in Appendix H. After these were implemented, the system was taken to Steve Graves Motors for the staff to test, which they accepted.

5.8 Production of User Manual

After the final changes were made to the system, a user manual for the software was produced as required by the employees of Steve Graves Motors. This had to be a non-technical guide on how to carry out the various functionalities of the system. In order to keep the manual as easy to follow as possible, step by step instructions were included to describe how to carry out the functions. Additionally, many screenshots were included to aid the user in getting a better understanding of exactly where to carry out the steps described. The user manual also included an installation section to describe to the user how to install the software. This is highly essential as there are various procedures that have to be carried out before the software can be used. After the manual was produced, it was taken to the users for them to have a quick browse through in order to identify whether they understand the descriptions on how to carry out the various functions. The feedback they gave after this review was positive, and they said that they understood the manual and it was not too technical. The user manual can be found in Appendix K.

6. EVALUATION

6.1 Introduction

This chapter discusses the evaluation of the software system developed for this project that was carried out. This is different to the user acceptance testing of the system carried out in the earlier stages of the project (as discussed in previous chapters). This is because the evaluation in this chapter consisted of a more detailed and thorough examination of the system's functionalities and usability by the employees of Steve Graves Motors after it was fully developed. An evaluation of the project itself including the methodology followed and the reflection on the project can be found in Appendix A.

6.2 Evaluation Criteria

In order to evaluate the system, a number of criteria had to be set out to identify how successful its outcome was. The following criteria were used to evaluate the system:

- ***Evaluation of User Requirements:*** to assess and measure the quality of the functionalities of the system that were implemented, as were required by the users.
- ***Usability Evaluation of System:*** to identify whether the usability of the system was good and easy to use according to Nielsen's [21] five attributes of usability.
- ***Comparison with Alternative Solutions:*** to compare the system produced with the other off-the-shelf software packages available, and determine the similarities and differences.
- ***Evaluation of Technology Used:*** to evaluate the effectiveness of the technologies that were used to develop the system.

6.3 Evaluation of User Requirements

It is highly essential that the functionalities implemented into the system are evaluated to identify whether they satisfy the users' requirements. After the system was fully implemented, it was given to Steve Graves Motors for the staff to use for a period of seven days. They would use the new system in parallel with their current system to carry out their usual everyday tasks and activities of the business. The evaluation consisted of giving the staff a form to fill at the end of the seven days in which they were required to give a score on how well each of their requirements are performed by the system, this would give a good measure of how well the requirements are satisfied. The criteria for evaluating each user requirement was whether the function meets the function's needs, how easy it is to perform, and if there were any errors encountered. The evaluation form with the results can be found in Appendix J. An analysis of the evaluation of the user requirements for the system are discussed below.

Manage Vehicle Stock Control: The feedback for the ability to manage vehicle stock control was generally good overall. However the users did mention that the format of entering the date in the Date First Registered field could have been made easier, rather than having to enter in the year first, then the month, and then the day. This format had to be implemented in this way because the MySQL database only accepts date formats to be yyyy-mm-dd.

Process Sale of Vehicle: The users agreed that the function to process the sale of a vehicle met its needs and that it was easy to use. However they mentioned that the method of having to print off an invoice by going to the HTML file where it is created, and then printing it off could have been made easier by printing it directly when the process sale button is clicked. The reason this was not done was because a method to implement such a feature could not be found.

Schedule Vehicle for Repair or Service: Meeting the function's needs of adding and viewing an appointment proved to be successful when evaluated by the users, which also returned no errors. However they mentioned that the ease of use when adding a new appointment was not the best because they cannot see a whole calendar format schedule indicating which dates are fully booked or available. It was initially anticipated that this sort of feature would be implemented, however after conducting research into how it could be developed, it was decided that it would be highly complex and extremely time consuming, which would impact the ability to implement the other requirements.

Search Vehicle: The functionality of searching a vehicle was regarded to be highly accurate as it returns no errors, and is also easy to use. However the users stated that it does not allow a search to be made within the additional notes of each vehicle. This was not implemented because the users did not mention this was required to be built into the search feature in the second iteration user testing.

Search Previous Sale: The users' evaluation on the ability to search a previous sale provided positive feedback, in particular the ease of use and the accuracy. They mentioned that if the search could be done with a customer's name as the input, the feature would be more desirable. However this was not an important issue just as long as the search could be made by the vehicle registration.

Calculate Total VAT Due: The systems ability to calculate the total VAT due also generated positive feedback in all three criteria in the user evaluation, and therefore fully satisfies the requirements of the user. The only criticism was the method of printing the VAT report, a problem similar to when printing a sales invoice and other printouts of the system.

Viewing Sales Performance of Business: The users agreed that this functionality satisfied their requirements very well, especially the ease of use. The users did mention that a feature whereby the sales figures could be plotted on a graph would increase the ability to make better comparisons of different dates selected. The ability to plot graphs in Java was thoroughly researched, however implementing this would have been extremely challenging and time consuming, which again would have had an impact on implementing the other required functionalities.

Invoicing Vehicle Scheduled with Work Carried Out: This functionality of the system proved to be easy to use and without any errors when evaluated by the users. However in meeting the function's needs, there could have been improvements. The users suggested that the fields to enter information about work carried out could have been slightly bigger. The reason this could not be implemented was because the maximum number of characters that can be stored in a column in the MySQL database is 255. Therefore a restriction had to be made on the length of the fields so that only a maximum of 255 characters could be entered by the user.

Login Access: The login access feature of the system provided positive feedback from the users in all three categories. This functionality satisfies the requirements of the users without any issues

Adding Image of Vehicle: This function proved to have met its needs and return no errors. However there was a suggestion for its ease of use. The users stated that when the window to choose the image file is opened, it does not automatically locate them to the directory where the images are required to be stored. They have to locate to the directory themselves. This was not included because it was unsure at the time where the directory of storing the images should be placed.

Viewing Sales Performance of Vehicles: Apart from its easy of use, this function got some negative feedback, in particular the slight errors associated with it. The users mentioned that the figures in the average profit column were not correct according to their needs. Rather than giving an average profit of each item based on its total stock, the users only wanted average profit figures to be calculated based on the number sold for that item. This would have been easy to implement, however it was not picked up during the user testing, and therefore could not be corrected.

Changing Company Details: The functionality of changing the company's details to cater for the system's printouts and different companies gained positive feedback in all criteria. The users were happy with this function as it met its needs, was easy to perform, and returns no errors.

User Manual: The user manual that was produced also proved to be successful in meeting the users' needs, being easy to follow, and highly accurate. This was probably mainly due to the care taken when writing the manual to ensure that it covers all aspects of the system, and was also kept non technical.

The enhancement to show how long a vehicle has been in stock for has been achieved by the *Days In Stock* column in the stock list table. The ability to give warnings if they have been in stock for a long time could not be implemented due to the fact that there was no time to create this. This feature was only a desirable requirement and therefore its exclusion does not affect the rest of the system.

In conclusion, it can be said that the user requirements of the system have been met to an acceptable degree. In almost all cases, each function meets its primary needs as was required when identified in the Analysis stage of the project, which was also easy to use. However there has been some slight criticism of the system in some areas, these were mainly suggestions to where an improvement about a certain aspect could have been made. The use of the system by Steve Graves Motors over the seven days allowed them to carry out a thorough evaluation of each of their requirements implemented in the system, which could not be thoroughly evaluated during the short user testing sessions. This is probably why the system's functionalities for the user requirements received a few negative comments, as the users got to use the system on a daily basis and therefore identify issues that were not noticed before. In terms of the user requirements identified in the Analysis stage and user testing feedback, their implementation into the system has proven to be successful.

6.4 Usability Evaluation of System

In addition to evaluating the user requirements implemented into the system, the usability of the system was also evaluated. This involved an evaluation of the software system as a whole, concentrating on aspects of it that are consistent throughout. The method used to evaluate the usability consisted of conducting two interviews; one was with the employees of Steve Graves Motors at the end of their seven day use of the system, and one with the staff of a different car sales company called RSC Auto Centre after they were given the chance to review the software system for a few hours. The reason behind including a different company was because this evaluation is based on the usability of the software, which is usually not specific to any company's requirements, but a more general aspect that can be evaluated by other potential users. The criteria used for this evaluation was based on Nielsen's [21] five attributes of usability, as discussed in the Background Research chapter. A full summary of the results of both interviews can be found in Appendix J, an analysis of these results are discussed below.

Learnability

The opinions on the ease of learning the system varied between the two companies. Steve Graves Motors believed the system was easy to learn after they got a bit of practice, whereas RSC Auto Centre found it difficult. This could have been down to two factors. Firstly, Steve Graves Motors used the system for a few days, so they had more time to get familiarised with it before they evaluated it. RSC Auto Centre only got to evaluate the usability for a few hours, and therefore had less time to learn how to use the system. Secondly, Steve Graves Motors mentioned that they used the user manual to help them learn the system, RSC Auto Centre did not refer to the user manual when using the system. This could have been because they did not have the time to go through the manual. From these results it can be seen that once the users become familiar to using the system, it can be easy to learn.

Efficiency

The feedback on the efficiency of the system was positive from both companies. They both stated that it initially took them time to get used to the system which slightly impacted efficiency. But once they used the functions a few times repeatedly, they found that they were able to carry out tasks quicker. This was probably mainly down to implementing the system to be as simple as possible by developing it to aid the user in carrying out tasks. For example when editing a vehicle or sale, the input fields were already populated with the current data about that item, and when processing the sale of a vehicle the vehicle tab was already populated with the vehicles details so they do not have to be entered again. This all helped in improving the efficiency of the system.

Memorability

The ease of remembering the system resulted in contrasting opinions between the two companies. Steve Graves Motors suggested that the system was easy to remember, whereas RSC Auto Centre believed that only the simple functions were easy to remember, but the more complex functions were not. The system was probably easy to remember due to the fact that it was designed in such a way that similar functions were to be carried out in similar ways to each other, for example the abilities to view and edit a vehicle or sale. It is highly likely that RSC Auto Centre found parts of the system difficult to remember again due to the fact that they only used the system for a few hours.

Errors

The system was successful at preventing errors when evaluated by the two companies. Both stated that they found no errors in any of the data that was entered into the system. Additionally they both mentioned that the system was good at preventing the possibility of accidental errors being made due

to the confirmation boxes it provides when deleting items. This is a very important factor of good usability, and due to this reason, this feature was implemented throughout the system.

Satisfaction

The opinions on the satisfaction of using the system varied between the two companies. Steve Graves Motors believed the system to be highly satisfactory up to the point of them considering whether to replace the old system with the new one, whereas RSC Auto Centre could not give a proper opinion on this aspect. The reason behind these contrasting opinions was again due to the fact that Steve Graves Motors got to use the system for a few days, and RSC Auto Centre only for a few hours. As a result, Steve Graves Motors got to see what the system was like on a daily basis, therefore they were able to make a more realistic decision for the company. They found the system to be subjectively pleasing because it only required them to store data once, unlike in the current system. In addition to this it enabled them to carry out tasks much quicker and with less effort, which would suit the employees and the business as it also offers more capabilities and functions.

In conclusion the usability of the system has gained much praise, with a few negative comments mainly down to the inability to get used to the system and its features quickly. As Steve Graves Motors were able to use the system for a duration of seven days, they became more familiar with how the system works, and hence gave more positive feedback about the usability of the system. RSC Auto Centre only got to use the system for a few hours and therefore could not get used to the system in the short time. This could potentially be a flaw in the usability of the system, an improvement could be made to make the system easier to adapt to, especially when the system would be required to be used immediately if deployed into the business. However the views of Steve Graves Motors were that once the system has been familiarised with, the usability of it is of fairly good quality.

6.5 Comparison with Alternative Solutions

There were three alternative software systems that were evaluated in section 2.1 of the Background Research chapter to see whether they would be suitable for Steve Graves Motors to satisfy their requirements. The evaluation resulted in neither of them being suitable for them to use. However it is important that the software system developed in this project is compared with these alternate systems in order to identify whether this system can meet the requirements of the market these software products exist in. The criteria for this evaluation consisted of which functionalities each system has, the interface, and the usability where applicable. These were all derived from information on their websites about the software and the available screenshots of them.

Autovendor is quite similar to the software developed in this project in its ability to store and sell vehicles. Both software allow the storage of an image and additional notes when adding a new vehicle, and the ability to print off an invoice when a vehicle is sold. However there are quite a lot of functionalities of the system developed in this project that are not included in Autovendor, these include the abilities to schedule and invoice a vehicle for a repair or service, search a vehicle, search a sale, view sales performance, and automatically calculate the total VAT due. The interface of Autovendor is good. The screens are not overpopulated with input fields, and it also uses tabs to separate the data, just like the system developed in this project. However it only includes the use of one main colour. The system developed in this project makes good use of contrasting colours and images across various areas of the system to make the interface look more professional.

When comparing Dragon2000 to the software developed in this project, Dragon2000 has many more functionalities including the ones developed for the system in this project. As well as stock control and processing a sale, these include the abilities to print various invoices, calculate profit and VAT, include four photos for a vehicle, search a vehicle, and personalise the invoices for each company. In addition to these features, Dragon2000 also has the abilities to write stock to web pages, a calendar style scheduling for a repair or service, service next due reminders, and mechanics performance ratings. The interface features are quite similar for both systems. Dragon2000 makes use of tabs and icons, however it does not include much colour on the screens, unlike the system developed in this project. The additional features of Dragon2000 could not be implemented into the system for this project due to its scope only consisting of a few months, whereas Dragon2000 has probably been in various development life cycles over a number of years with many developers working on the system.

A free 30 day trial period for Resoco CarKey was obtained and evaluated, paying much attention to its usability. The functionalities of CarKey and the software developed in this project are very similar. They both have the abilities to stock and sell vehicles, search a vehicle, display a photograph of a vehicle, print invoices, store monthly finance details, and viewing sales performance. However the software developed in this project also includes the abilities to calculate the total VAT due, and schedule and invoice a vehicle for a repair or service. When testing CarKey, the usability of it was not of the best quality. Firstly the interface was not too appealing, there was no use of colour except on the text fields, the labels were very small and difficult to see, and the screens were overcrowded with too much data. There was also no separation of data entry, so vehicle details and pricing information have to be entered on the same screen when processing a sale, unlike the system developed in this project, this makes the usability of CarKey unpleasant.

6.6 Evaluation of Technology Used

The technology used in this project proved to be successful at meeting the requirements of developing the system. The MySQL database was easy to use due to its simple command line interface, and it had the capabilities to perform all the queries required from the system to manage the data stored. Java, with its capabilities to create a GUI using its Swing framework components, also allowed a good front-end software application to be developed. Apart from some features of the system that could not be implemented due to the high complexity and amount of code required to be written in Java, creating the software application was comfortable mainly due to the previous experience of creating applications in Java. Using Eclipse to write the Java code to create the application proved to be highly successful due to its many features in helping the developer. This included highlighting errors in syntax, pointing out code that was missing, and pointing out exceptions thrown for errors at runtime.

6.7 Possible Further Extensions

There are many further improvements and extensions that could be added to the system developed in this project. These have been identified from the evaluation of the system carried out by the users, and from the evaluation of the alternative systems. These main extensions include the following:

- ***Give warnings if vehicles been in stock for long time:*** this was initially a required enhancement of the system whereby vehicles in the stock table would be highlighted in different colours, this could not be implemented due to time constraints.
- ***Print invoices directly from system:*** rather than writing to a HTML file which can be printed off, the system could allow a printout to be made directly from its print buttons on the screens.
- ***Calendar format to schedule vehicle:*** when scheduling a vehicle for a repair or service, a calendar style scheduling system could be developed to view all dates booked and available.
- ***Graphs to show sales performance:*** in order to better view and compare sales performance between two set of selected dates, the figures could be plotted and displayed on a graph.
- ***Send service next due reminders:*** the system could include a feature to automatically send reminders to customers that their next service is due, as included in Dragon2000.
- ***Implement backup strategy:*** a system backup feature could be included to backup the data stored on the database in case of any possible system or computer hardware failures.
- ***Assistance in learning system quickly:*** due to the negative feedback for ease of learning, an assistance feature could be added on the system to help users learn it quickly.
- ***Integrate with website:*** an integration can be made with a web based system whereby when stock is updated, the company's website is updated with the new stock for customers to see.

7. CONCLUSION

The main challenge of this project was to develop a software application that includes the various functionalities required by Steve Graves Motors, and to make the usability of this software to be of good quality by cohering to Nielsen's [21] five usability attributes. The aim of the project was to identify the problems encountered by Steve Graves Motors in the running of their business with the current system in place, and then to analyse the requirements, design, and implement a software application to provide a solution to their problems through a number of iterations. In addition to this it was highly important to test the system at various stages of the project, and then to evaluate it at the end in order to identify whether the system has been built to satisfy the users' requirements.

All the minimum requirements and further enhancements required from the system identified in Analysis stage of the project were implemented into the software application, apart from the ability to give warnings for vehicles in stock for a long time. During the testing and evaluation stages, the users accepted the fact that these functionalities have been built to their initial specified requirements, despite suggesting some possible improvements to them afterwards that were not identified during the user testing stages. In addition to this, the system produced also includes many functionalities that other alternative solutions do not offer. Finally the employees were also satisfied with the usability of the system, believing it to be very efficient once they have got used to using it after a short while.

The software application developed in this project was not without its criticisms. The employees of Steve Graves Motors found a few minor negative aspects about the functionalities of the system during the evaluation stage, and RSC Auto Centre who also evaluated the system found a few difficulties with its usability during the initial stages of their evaluation. Therefore there is much further work that can be done to the system in order to improve its usability, and add further functionalities to it to satisfy the requirements of many other car sales and servicing companies.

It can therefore be said that the project has been of some success. The objectives that were initially set out have completed to a high level of standard. The minimum requirements and further enhancements of the system have been implemented to an acceptable level, which also provide a solution to the problems encountered in the current system for the employees of Steve Graves Motors. The final success of the project now depends on whether Steve Graves Motors decide to deploy the system for use in their company, this is something that they have seriously considered during the final stages of their evaluation of the system.

REFERENCES

- [1] Avison, D & Fitzgerald, G. (2003), *Information Systems Development: Methodologies, Techniques and Tools*, 3rd Edition, McGraw-Hill.
- [2] Cadle, J & Yeates, D. (2004), *Project Management for Information Systems*, 4th Edition, Prentice Hall.
- [3] Hughes, B & Cotterell, M. (2002), *Software Project Management*, 3rd Edition, McGraw-Hill.
- [4] Chester, M & Athwall, A. (2002), *Basic Information Systems Analysis and Design*, 1st Edition, McGraw-Hill.
- [5] Sommerville, I. (2004), *Software Engineering*, 7th Edition, Pearson/Addison-Wesley.
- [6] Curtis, G & Cobham, D. (2005), *Business Information Systems*, 5th Edition, Prentice Hall.
- [7] Resoco CarKey, URL: <http://www.resoco.co.uk/> [Cited: 27/11/2006]
- [8] Dragon2000, URL: <http://www.dragon2000.co.uk/> [Cited: 27/11/2006]
- [9] AutoVendor,
URL: <http://www.littlesystems.co.uk/Pages/Products/AutoVendorHome.html>
[Cited: 28/11/2006]
- [10] Drayton, P, Albahari, B & Neward, T. (2003), *C# in a Nutshell*, 2nd Edition, O'Reilly.
- [11] MSDN. (2006), *Learn C#*, Microsoft Corporation,
URL: <http://msdn2.microsoft.com/en-gb/vcsharp/aa336766.aspx#framework>
[Cited: 11/12/2006]
- [12] MSDN. (2006), *Database Access (C# vs. Java)*, Microsoft Corporation,
URL: <http://msdn2.microsoft.com/en-us/library/ms228366.aspx> [Cited: 11/12/2006]

- [13] Roman, S, Petrusha, R & Lomax, P. (2002), *VB.NET Language in a Nutshell*, 2nd Edition, O'Reilly.
- [14] MSDN. (2006), *Learn Visual Basic*, Microsoft Corporation,
URL: <http://msdn2.microsoft.com/en-gb/vbasic/ms789086.aspx>
[Cited: 11/12/2006]
- [15] Wigglesworth, J & McMillan, P. (2004), *Java Programming: Advanced Topics*, 3rd Edition, Thomson Course Technology.
- [16] Choudhari, P. *Java Advantages and Disadvantages*, ArizonaCommunity.com,
URL: http://arizonacommunity.com/articles/java_32001.shtml
[Cited: 12/12/2006]
- [17] Atzeni, P, Ceri, S, Paraboschi, S & Torlone, R. (1999), *Database Systems: Concepts, Languages and Architectures*, 1st Edition, McGraw-Hill.
- [18] Sinclair, R. (2000), *From Access to SQL Server*, 1st Edition, Apress.
- [19] Butcher, T. (2003), *Sams Teach Yourself MySQL in 21 Days*, 2nd Edition, Sams.
- [20] DuBois, P. (2005), *MySQL: the definitive guide to using, programming, and administering MySQL 4.1 and 5.0*, 3rd Edition, Sams.
- [21] Nielsen, J. (1993), *Usability Engineering*, Academic Press.
- [22] Storkel, S. (2002), *An Introduction to the Eclipse IDE*, O'Reilly Media,
URL: <http://www.onjava.com/pub/a/onjava/2002/12/11/eclipse.html>
[Cited: 12/12/2006]
- [23] Mono. (2006), URL: <http://www.mono-project.com/> [Cited: 15/12/2006]
- [24] Wise, W. (2004), *Compatibility - A Few Missing Pieces*, ASPAlliance.com,
URL: <http://aspalliance.com/387#Page3> [Cited: 15/12/2006]

- [25] Avgerou, C & Cornford, T. (1998), *Developing Information Systems: Concepts, Issues and Practice*, 2nd Edition, Palgrave.
- [26] Kendall, K & Kendall, J. (2002), *Systems Analysis and Design*, 5th Edition, Prentice Hall.
- [27] Bennett, S, McRobb, S & Farmer, R. (2006), *Object-Oriented Systems Analysis and Design Using UML*, 3rd Edition, McGraw-Hill.
- [28] Maciaszek, L. (2005), *Requirements Analysis and System Design*, 2nd Edition, Addison-Wesley.

APPENDIX A – Project Reflection

Conducting this project has been one of the most important and challenging aspects of my educational history. Despite some of the difficulties I faced, I have enjoyed this project as it has achieved my personal objectives that I had expected from a final year project. These include further developing my technical skills in programming in Java, databases with MySQL, skills in following a software development life cycle, and skills in timing and meeting deadlines. I believe that these will all prove to be extremely valuable in the future during my career within the computing industry.

Evaluation of Methodology Followed

There were many advantages to following the prototyping methodology for this project. This mainly consisted of the ability to gain user feedback between various stages of developing the software application in order to see whether their initial requirements have been met, and gaining feedback on any additional items they would like modified or improved. This proved to be highly successful at first, as the staff of Steve Graves Motors gave many ideas for improving the system at the next iteration of developing the system. However these testing sessions with the user only lasted for a few hours, and therefore they could not make a good enough evaluation of the system at that time. When the system was given to Steve Graves Motors for seven days during the evaluation stage of the project, they found a few drawbacks to it and possible further improvement that they had not identified during the user acceptance testing stages. The result of this shows that a system cannot be fully tested until it is put into a proper trial period whereby it is used to conduct real business activities. This is because users only get to know and feel what a system is really like when they use it regularly and during important times. A valuable lesson that can be learnt from this is that rather than waiting until the evaluation stage to see whether a software application has been developed to fully satisfy the user's requirements, the system should be thoroughly tested at the end of each iteration by allowing the user to use the system for more than just a few hours. By doing this, any issues with the system could be addressed earlier and implemented before fully finishing it and allowing the user to evaluate it. In addition to this, the system could have been evaluated by more potential users, in order to identify whether the functionalities implemented could possibly satisfy other car sales and servicing companies. By doing this, the usability of the system could also have been evaluated further. This could have resulted in other possible users to have found it to be easy or difficult to use, just as RSC Auto Centre thought it was difficult to learn during the initial stages of their evaluation.

Things That Went Well / Badly

Overall, there was a mixture between the things that went well and badly in the project. One of the things that went well was the timing of developing the system, this was achieved by starting the implementation of the system early and investing a huge amount of time into it. As a result, the minimum requirements and further enhancements were implemented just before the progress meeting. However the schedule to write the report did not go very well. Initially it was thought that this would not be a problem as the implementation section of the report will be written up as the implementation of the system is taking place. But this was not the case in the end, and the implementation section in the report was written after the system was fully developed due to concentrating the time available on developing the system. As a result the schedule to write the report was falling behind and extra work had to be put in to ensure that the report will be completed on time.

Lessons Learnt From Completing Project

There were many lessons that were learnt during the accomplishment of the project. These include things that I would do different if conducting another similar project, and they are lessons that future students conducting a final year project could learn.

Involve the Users: involving the users is very important in any project. They set the requirements of the project which if not included, there will not be much to gain from the project. My experience of involving users was valuable, however I would advise to include them as much as possible, especially at the testing stages so that any modifications and improvements are identified earlier.

Do not underestimate time required to write report: writing a report of extremely high quality as required for a final year project takes a huge amount of time, which at first is not realised. Therefore make sure that for every spare time available during University, write up a part of your report.

Choose a project of interest: choosing a project that involves something you are not interested in or unfamiliar with can be de-motivating, and as a result could affect your chances of doing a good job. Choose a project that you will be interested in and you may find it more interesting and exciting than you initially would have thought, just like I did for this project.

Try solving a problem yourself first: by solving a problem you encounter in the system you develop yourself first, you will find that you will learn a lot better from this, and next time you will know exactly what to do. If you cannot solve the problem yourself, then try to seek help from someone else.

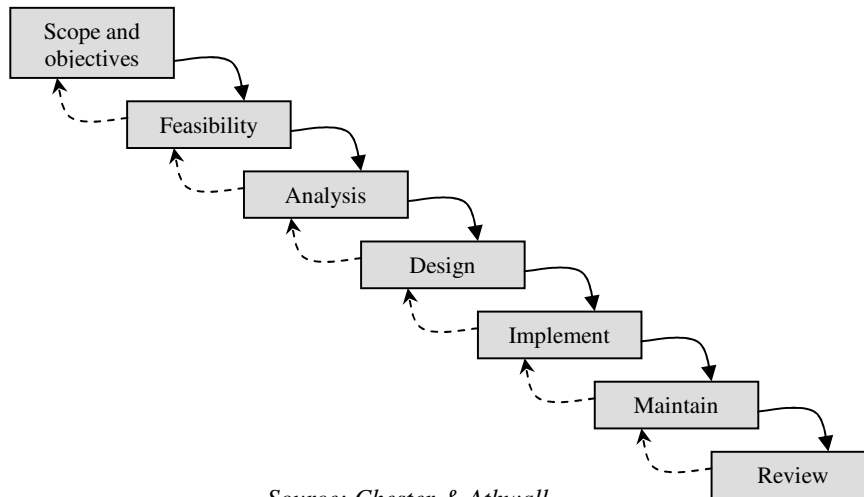
APPENDIX B – Project Schedule

Initial Schedule		Semester 1										Winter Break				Exams		Semester 2						Spring Break				Sem. 2						
		Month		October-2006				November-2006				December-2006				January-2007				February-2007				March-2007				April-2007						
	Week	2	3	4	5	6	7	8	9	10	11							1	2	3	4	5	6	7	8					9	10			
Task																																		
Aim & Objectives																																		
Background Research																																		
Methodology																																		
Alternative Solutions																																		
Development Tools																																		
Feasibility Assessment																																		
Analysis																																		
Gather Requirements																																		
Requirements Analysis																	EXAM PERIOD																	
Business & System Modeling																	EXAM PERIOD																	
Design																																		
Database Design																																		
System Design																																		
Implementation																																		
Database Development																																		
Interface Development																																		
Functionality Development																																		
Testing																																		
Unit Testing																																		
Integration Testing																																		
Deployment Testing																																		
Evaluation																																		
Report Writing																																		

Revised Schedule		Semester 1										Winter Break				Exams		Semester 2										Spring Break				Sem. 2		
	Month	October-2006					November-2006					December-2006					January-2007					February-2007					March-2007				April-2007			
	Week	2	3	4	5	6	7	8	9	10	11							1	2	3	4	5	6	7	8					9	10			
Task																																		
Aim & Objectives																																		
Background Research																																		
Methodology																																		
Alternative Solutions																																		
Development Tools																																		
Usability																																		
Analysis																																		
Feasibility Assessment																																		
Assess Current System																																		
Gather Requirements																																		
Requirements Analysis																																		
First Iteration																																		
System Design																																		
Implementation																																		
Unit and User Testing																																		
Second Iteration																																		
Implement Changes																																		
System Design																																		
Implementation																																		
Unit and User Testing																																		
User Manual																																		
Evaluation																																		
Report Writing																																		

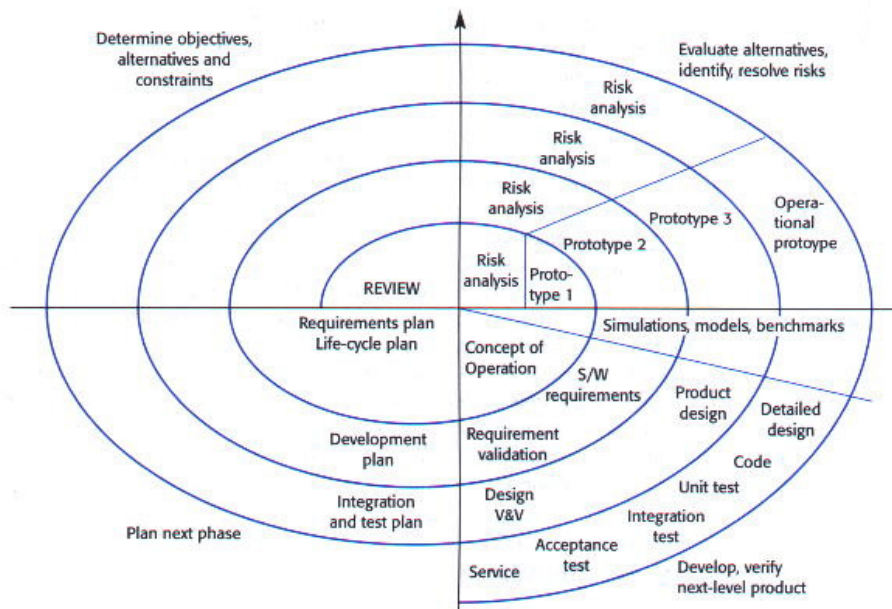
APPENDIX C – Methodology Models

Waterfall Model



Source: Chester & Athwall

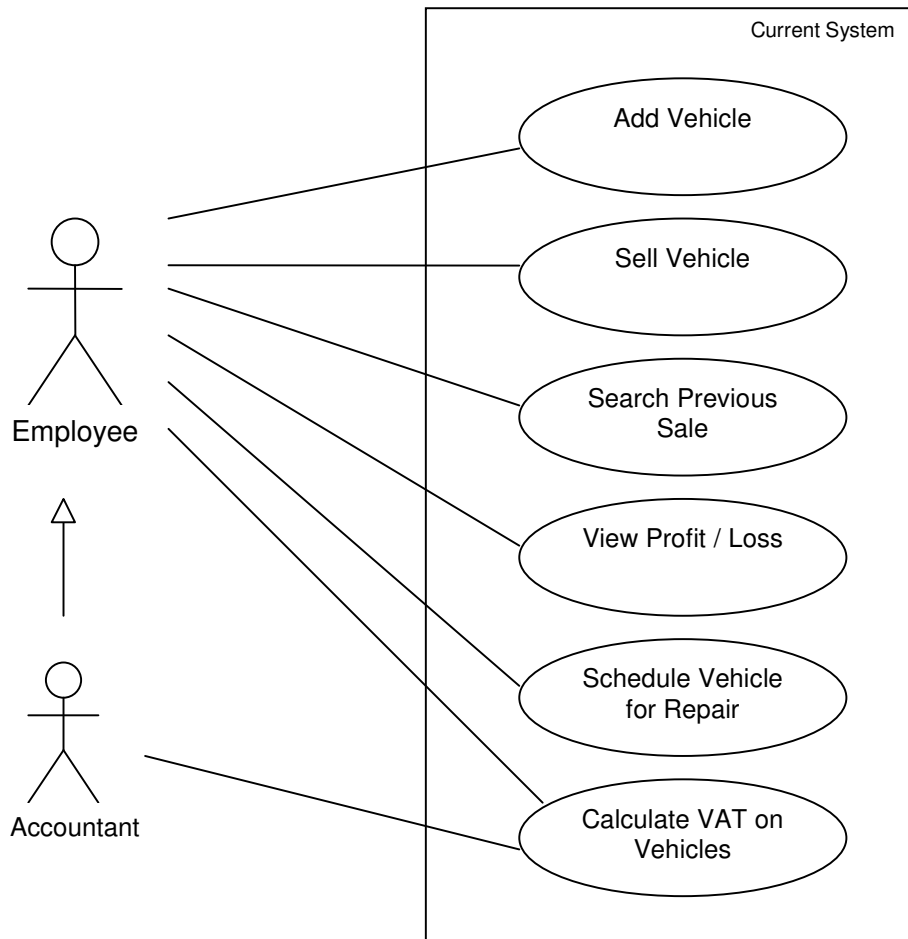
Spiral Model



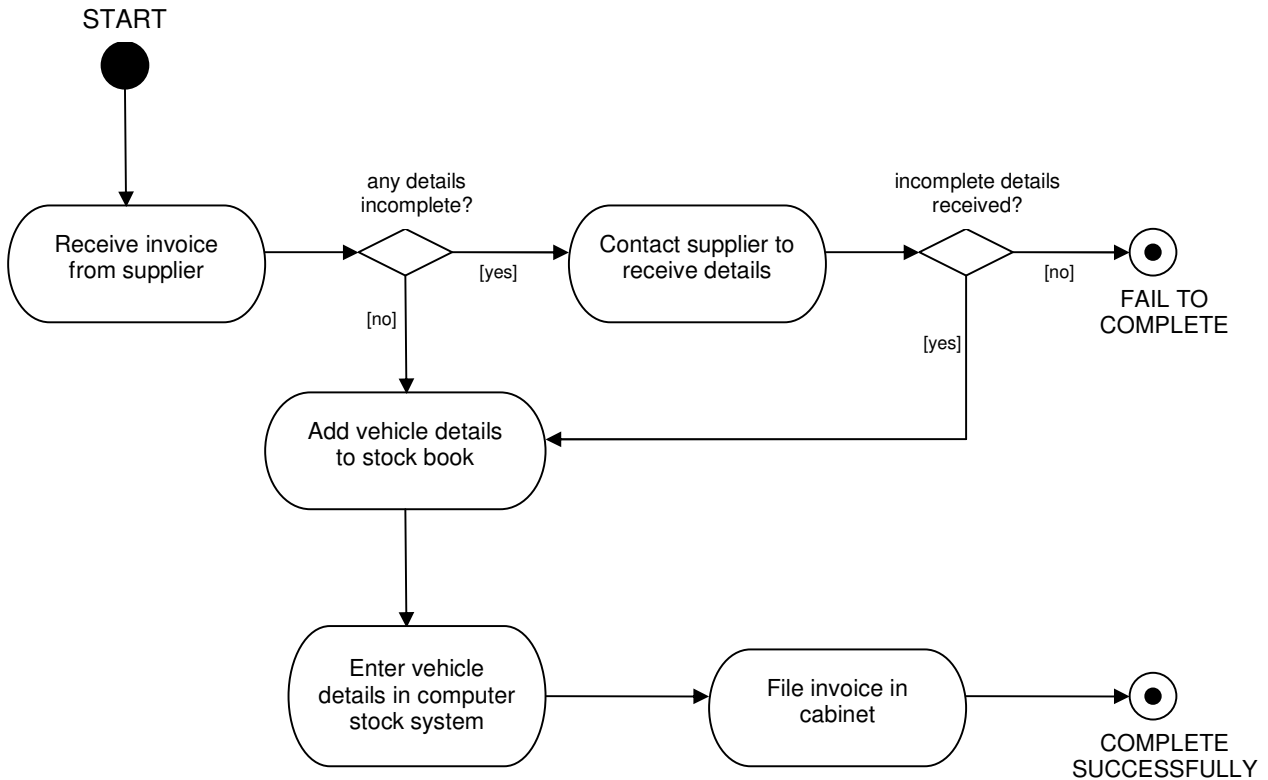
Source: Sommerville [5]

APPENDIX D – Analysis

Use-Case Diagram for Current System

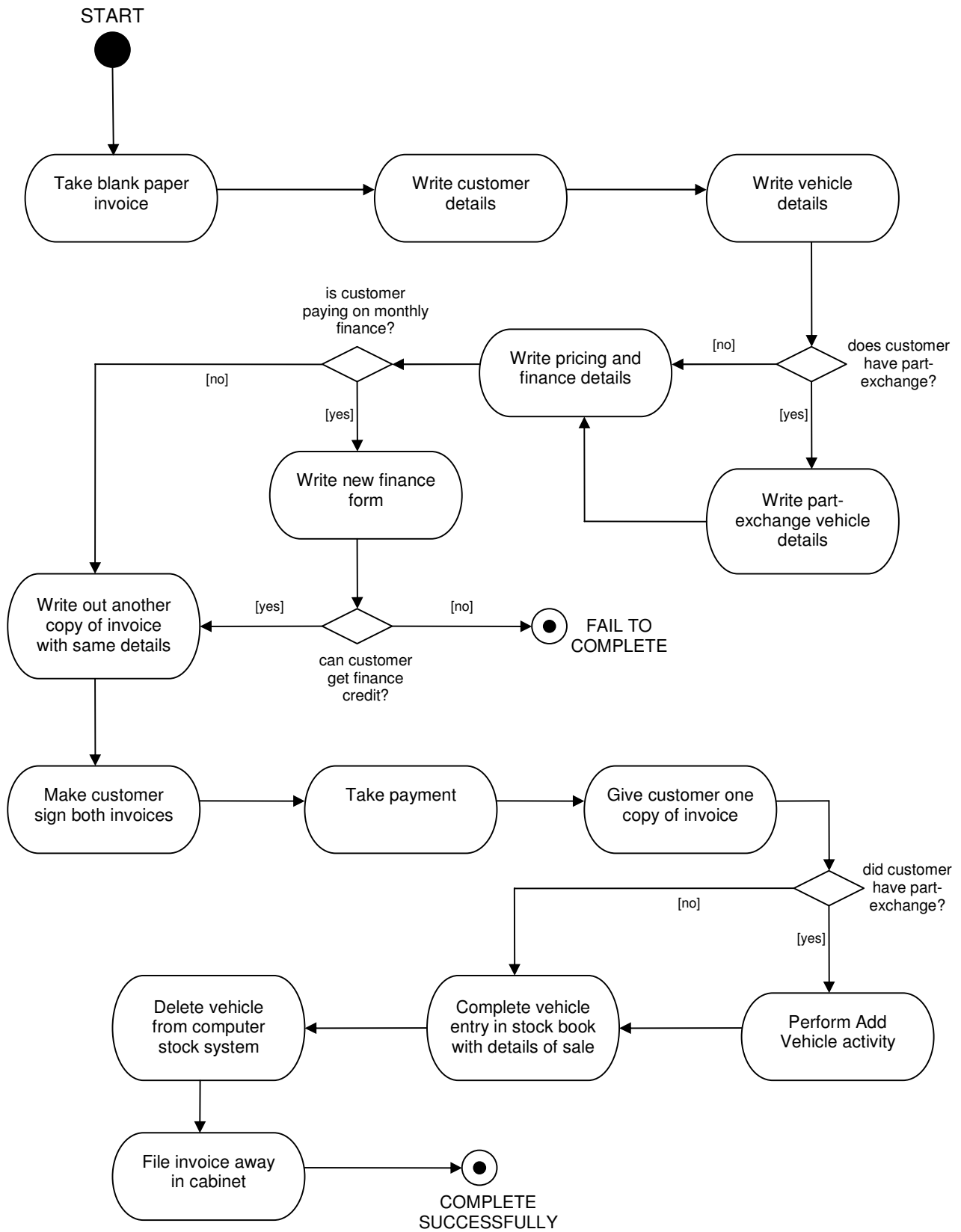


Activity Diagram for Add Vehicle Use-Case



The process begins when an invoice is received from the supplier of a vehicle, and the details are checked to see whether all the relevant information is present. If not, then the supplier is contacted to receive the incomplete details, and once they are received the vehicles details from the suppliers invoice are entered into the stock book as normally would be done if all required details were present on the invoice in the first place. The same vehicle details are then entered into the current computer stock system, and finally the invoice is filed away.

Activity Diagram for Sell Vehicle Use-Case



The Sell Vehicle process begins when the vehicle to be sold, customer's, and the part-exchange vehicle details (if necessary) are written down on a new blank paper invoice. Next the pricing and finance details are written, and if the customer is purchasing the vehicle on monthly finance then a separate finance form is filled out. If confirmation that the customer cannot obtain credit is received, then they are unable to purchase the vehicle and therefore the transaction fails. Otherwise if they can obtain credit then the process continues whereby the employee fills out another invoice with the same details, this is so that the customer and the company can both keep a copy of an invoice. Next the customer is asked to sign both copies of the invoice, and the payment is made by the customer, whether it is just a deposit or a full payment. They are then given their copy to take away. The final part of the sale process is if the customer did a part-exchange on their old vehicle, then the employee would have to go through the whole process of the Add Vehicle use case in order to add the vehicle to the company's stock list. After this is completed, the entry of the vehicle sold in the stock book is updated by filling in the details of the sale such as the customer details and the price it was sold. Finally the vehicle entry is deleted from the computer stock system and the invoice is filed away in a cabinet. The sale of a vehicle is now regarded as being completed successfully.

Description of other Use-Cases

For the Search Previous Sale use-case, I was given the chance to be involved in searching for the details of a sale that was completed a few months ago. I was given the registration number of a vehicle that was sold, and using this I had to go through the stock book looking at each sale entry to try and find a match with the registration number I was given until the correct sale entry was found. For the View Profit/Loss use-case, one of the employees showed me how its process is normally carried out. This is done by going through the stock book and calculating the total profit made for a selected month by adding up the profit made from each vehicle sold in that month. This figure is then noted down on a sheet with all other figures that were calculated for previous months. The figure calculated for that month is then compared to the profit made from the previous months. The process involved for the Schedule Vehicle for Repair use case simply involves an employee making a note on paper for when the customer would like to bring their car in. This is then kept in the office and the information about the schedule is retrieved when a customer has brought in their car on the day. For the Calculate VAT on Vehicles use case, an employee showed me how the accountant normally calculates the figures. In the stock book for each sale that is complete, the accountant is required to calculate the VAT of the profit made on selling the vehicle excluding any additional expenditure occurred on it. This is done every few months and the accountant is required to go through and calculate the VAT on each vehicle along with the total VAT for that period.

Invoice used for Processing Sale of Vehicle

USED VEHICLE INVOICE Steve Graves Motors		INVOICE No.....
SOLD BY :		
Unit 1A Huddersfield Rd ,Birstall WF17 9AY TEL:01924 475382 FAX:01924 475282		
SOLD TO : (Name)..... (Address)..... Post Code..... Telephone No.....		
Particulars of car:- Make..... Make or Type..... Chassis No..... Registration No.....		Date first registered in U.K. Engine No..... Colour..... Speedometer reading..... * Not Guaranteed (see footnote)
I/we declare that I/we am/are the buyer(s) of the car described above I/we acknowledge receiving delivery of the car described above		
Date..... Signature(s).....		
DEDUCT: Allowance for used vehicle which I/we certify conforms to the following description (is my/our own property and free from any lien or legal encumbrances/is subject to H.P. as stated).		Price £ : : Net allowance £ : Deposit £ : Due From Cust. _____
Make..... Reg No..... Chassis..... First reg.....	Model..... Colour..... RFL expire date..... Speedometer reading.....	
Declaration by the seller:- Input tax deduction has not been and will not be claimed by me in respect of the car sold on this invoice.		
Date..... Signature.....		
* The speedometer reading is entered as an aid to establish mileage covered (from our delivery date to you) in the event of any future warranty claim. It is not suggested that it is correct mileage for this vehicle from the date of manufacture and does not form in any way a condition of sale.		
VAT reg No. 686 4859 66		

Interview with Employees of Steve Graves Motors

1) From the processes carried out in the current system only, what features would you like to be included in the new system and which are the most important?

- *The most important are:*
 - *The ability to add a vehicle to the company's stock, store its details on a computer system, and edit the details of the vehicle when required.*
 - *To process the sale of a vehicle on a computer system and print out the invoice, store the details of the sale so that its retrieval can be done instantly in future, and reduce the number of times data has to be inputted.*
 - *The ability to store information about the booking of a vehicle on specific dates.*
- *For the system to automatically calculate profit and losses from the sales figures entered when processing the sale of a vehicle and view them when required. Also it should be able to compare figures with previous months.*
- *For the VAT due on each sale and the total VAT due every few months to be automatically calculated by the system, and not having the employee calculate it manually.*

2) For adding a vehicle to the stock, what data/information do you require to be stored by the new system?

- *Its registration number, make, model, version, engine size in cc, number of doors, colour, body style, fuel type, transmission, date registered, chassis number, mileage, price vehicle was bought at, price to be sold, date added to stock.*

3) For processing and storing details of the sale of a vehicle, what additional data other than that from the current invoice do you require to be included in the new system?

- *The details of the vehicle should be automatically retrieved from its entry in the stock list so they don't have to be entered again.*
- *Additional information about the customer would be good such as email address so contacting them would be easier.*
- *When the part exchanged vehicle details are entered during a sale, the system should add this vehicle to the stock automatically.*
- *If customers are buying on monthly finance then data required to be stored will be the finance company, amount of loan, duration of loan, payment per month.*

4) For viewing profit and loss and comparing them with previous months, exactly what information should the new system show?

- *It should be able to calculate and show the profit made each month by calculating the profit made from each vehicle, for the new system the figures should be retrieved from the buying and selling values that were entered when adding and selling a car.*

5) For calculating the VAT due, exactly what information do you require to be shown by the new system?

- *It should calculate the VAT from the profit made from each sale by taking its buying and selling price. Total VAT is normally calculated and paid every few months so the figure must be from the sales made in the previous few months.*
- *When the VAT is taken every few months, for each vehicle sold in that period the accountant requires each vehicle's registration number, supplier details, customer details, profit made on that vehicle and VAT due on that vehicle.*

6) For booking of a vehicle for a repair or service, what data or information do you require to be stored by the new system?

- *Customer name, contact phone number, car registration number, make, model, engine size, date to be booked, what requires to be done.*

7) Other than the processes normally carried out in the current system, what other functionality and features would you like to be included in the new system?

- *Rather than having to search through the stock book for a previous sale that was made, the new system should be able to retrieve the details of a previous sale by entering the registration number in the system.*
- *To search the stock for a specific type of vehicle when the customer is not sure exactly what car they want, but they want a vehicle with specific specifications.*
- *Allow an image of a car to be added to the system when a new car is added to the stock list.*
- *It would be good if the new system can store information about work that was carried out on a vehicle after it was booked in for a repair or a service.*
- *Make sure the system is secure so non staff members cannot access the system.*
- *It would be a good feature if sales performance of vehicles can be viewed by their make or body type to see which are the most common and best selling ones.*

8) For searching the stock for a specific vehicle, what specifications of the vehicle would customers usually want to search for?

- *Customers normally want to search specifications of a vehicle similar to the ones already being inputted and stored in the system for the add vehicle functionality.*

9) For work carried out on a car, what information would you like to show on the invoice?

- *A list of what work was carried out and what parts fitted, the price of parts that were used.*
- *It should also show the total cost of the work carried out.*

10) How would you like the interface and usability of the new software application to be?

- *The software should be easy to use and simple as possible but also able to carry out the functionality required.*
- *Its interface should be clear, not too much colour or decoration, a good size for the window, text, buttons, boxes, etc.*
- *The usability should be so that menus are easy to access, processing the activities required should be able to be done with minimal effort required, it should not let the user have to browse through too many menus and windows to carry out a process.*

11) Do you think the functions of sending out reminders to customers that their service is due, and personalising a company's details on the system would be useful?

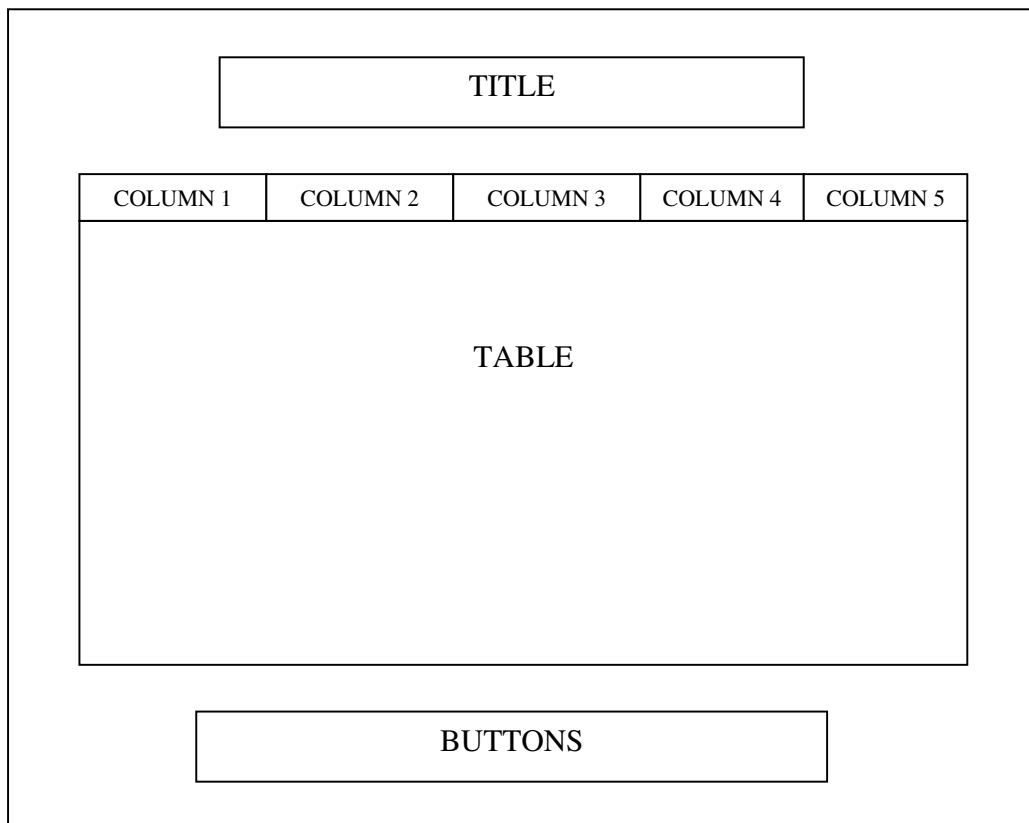
- *Sending out reminders that a service is due isn't really that necessary because customers normally don't forget this. The feature would not really be used that much.*
- *Changing a company's details may prove to be useful especially if the company is going to be relocated, so the address and contact details would have to be changed.*

12) Is there anything else you would require from the new system?

- *A user manual would be needed to refer to how some parts of the software work especially when showing new users or staff.*

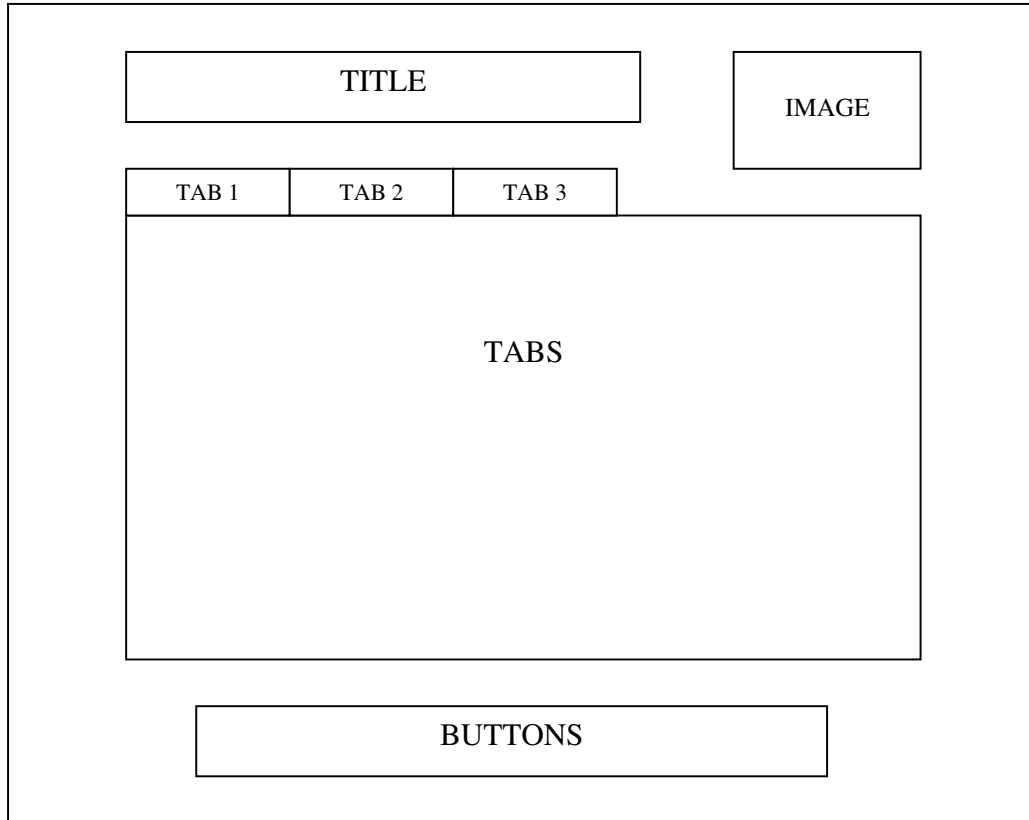
APPENDIX E – Design of First Iteration

Figure E1 - Design Layout of Screen with Table



This is what a screen which will contain a table is intended to look like. The title, column names, and buttons will vary according to what screen it is and what functionality it provides. The background colour of the screen will be light blue, and the background colour of the table will be light brown. The colour of all the text on this screen will be black.

Figure E2 - Design Layout of Screen with Tabs



This is what a screen which will contain tabs is intended to look like. The title, tab labels, and buttons will vary according to what screen it is and what functionality it provides. It was decided that tabs will be used to separate the data because Maciaszek [28] suggests that tabs are “*useful when the amount of information to be displayed in a secondary window exceeds the window’s ‘real estate’ and the subject matter can be broken logically apart into information groups*”. The background colour of the screen will be light blue, and the background colour of the tabs will be light yellow. An image related to the car industry will give the screen a professional look, however it is not intended to add images until the second iteration.

Design Layout of Tab

TAB	
Field Label 1	
Field Label 2	
Field Label 3	▾
Field Label 4	
Field Label 5	▾

This is what the layout of a tab is intended to look like. The Field Labels will simply state what data is required to be inputted into the input field next to it. An input field will consist of a normal text field where the user can enter text into it, or a drop down combo box where the user will be able to select from a list of options what data they would like to set for that field. This will vary depending on what is appropriate for that field.

Design of Vehicle Stock Control Functionality

The main screen for the stock control functionality will be a design layout of a screen with a table as shown in Figure E1 above which will show all the vehicles currently in stock. The columns to be included are *Registration Number*, *Make*, *Model*, *Version*, *Price*, *Days in Stock*. The buttons to be included on this screen to navigate to another place in the system include:

- *Add New Vehicle*: go to screen where user can add a new vehicle.
- *View Vehicle Details*: view all the details of a vehicle selected from the table.
- *Main Menu*: go back to main menu.

The screens for the functionality of adding, editing, or viewing a vehicle will be a design layout of a screen with tabs as shown in Figure E2. The different tabs on each of the screens will be *Vehicle* and *Supplier*. The *Vehicle* tab will include the fields *Registration Number*, *Make*, *Model*, *Version*, *Engine Size*, *Colour*, *Number of Doors*, *Body Style*, *Fuel Type*, *Transmission*, *Mileage*, *Date First Registered*, *Chassis Number*. The *Supplier* tab will include the fields *Previous Supplier*, *Name*, *Address*, *Postcode*, *Phone*, *Email*, *Date*, *Purchase Price* (although not to be included on view vehicle screen), *Selling Price*. The buttons to be included on the Add Vehicle screen to navigate to another place or to carry out a function include:

- *Add Vehicle*: add vehicle details into the database.
- *Cancel*: go back to main stock control screen.

The buttons to be included on the Edit Vehicle screen to carry out a function include:

- *Update Vehicle*: update details of vehicle in database with the data entered in fields.
- *Cancel*: go back to main stock control screen.

The buttons to be included on the View Vehicle screen to navigate to another place or to carry out a function include:

- *Edit Vehicle*: go to screen where user can edit the vehicle.
- *Sell Vehicle*: go to screen where user can process the sale of the vehicle.
- *Delete Vehicle*: delete vehicle from the database.
- *Back*: go back to main stock control screen.

Design of Process Sale of Vehicle Functionality

The screen for the functionality of processing the sale of a vehicle will be a design layout of a screen with tabs as shown in Figure E2. The different tabs to be included will be *Vehicle*, *Customer*, *Part-Exchange*, *Pricing*. The fields within the Vehicle and Part-Exchange tabs will be the same format as the Vehicle tab for when adding a new vehicle, however the fields in the Vehicle tab will be populated with the vehicles details obtained from the database. The Customer tab will include the fields *Title*, *First Name*, *Surname*, *Address*, *Postcode*, *Phone*, *Email*. The Pricing tab will include the fields *Date of Sale*, *Payment Method*, *Finance Company*, *Loan Amount*, *Duration*, *Monthly Payment*, *List Price*, *Final Price*, *Part Exchange Value*, *Amount to Pay*, *Amount Paid*, *Amount Due*. The buttons to be included on this screen include:

- *Process Sale*: add sale details to the database and write invoice to html file to print off.
- *Cancel*: go back to vehicle details screen.

The main screen for viewing previous sales will be a design layout of a screen with a table as shown in Figure E1. The columns to be included are *Registration Number*, *Vehicle*, *Customer*, *Date Sold*. The buttons to be included on this screen include:

- *View Sale Details*: view all the details of a sale selected from the table.
- *Main Menu*: go back to main menu.

The screen for viewing a sales full details or editing a sale will be a design layout of a screen with tabs as shown in Figure E2. The tabs to be included will be *Vehicle Sold*, *Supplier*, *Customer*, *Vehicle Part-Exchanged*, *Pricing*. The fields to be included in these tabs will be the same format as previously stated. The buttons to be included on the View Sale screen include:

- *Edit Sale*: edit the details of the sale.
- *Delete Sale*: delete vehicle, pricing, and customer details from database for that sale.
- *Back*: go back to main screen of previously sold vehicles.

Design of Schedule Vehicle for Repair or Service Functionality

The main screen for the Vehicle for Repair or Service functionality will be a design layout of a screen with a table as shown in Figure E1 above which will show all the vehicles currently scheduled for work to be carried out. The columns to be included are *Registration Number*, *Vehicle*, *Customer*, *Work Type*, *Date*. This screen will also have a date selection box where the user can select a date and view the appointments or check the availability for that date from the table. The buttons to be included on this screen to navigate to another place in the system include:

- *New Appointment*: go to screen where user can add a new appointment.
- *View Appointment*: view all the details of an appointment selected from the table.
- *Main Menu*: go back to main menu.

The screens for the functionality of adding, editing, or viewing an appointment will be a design layout of a screen with tabs as shown in Figure E2. The different tabs on each of the screens will be *Customer & Vehicle* and *Work Required*. The Customer & Vehicle tab will include the fields *Customer Name*, *Phone*, *Registration Number*, *Make*, *Model*, *Engine Size*. The Work Required tab will include the fields *Work Type*, *Date*, *Work Required*. The buttons to be included on the Add Appointment screen to navigate to another place or to carry out a function include:

- *Add Appointment*: add appointment details into the database.
- *Cancel*: go back to the main scheduling screen.

The buttons to be included on the Edit Appointment screen to navigate to another place or to carry out a function include:

- *Update Appointment*: update details in the database with the data entered in fields.
- *Cancel*: go back to the main scheduling screen.

The buttons to be included on the View Appointment screen to navigate to another place or to carry out a function include:

- *Edit Appointment*: go to screen where user can edit the appointment details.
- *Cancel/Delete*: cancel the appointment and delete the details from the database.
- *Cancel*: go back to the main scheduling screen.

APPENDIX F – Testing of First Iteration

Unit Testing for Implementing Vehicle Stock Control

TEST	EXPECTED RESULT	ACTUAL RESULT	COMMENTS
Enter non-alphanumeric characters in 'Registration' field on Add Vehicle screen	Do not allow the non-alphanumeric characters to be entered	As expected	
Enter non-numeric characters in 'Engine Size' and 'Mileage' fields on Add Vehicle screen	Do not allow the non-numeric characters to be entered	As expected	
Enter any characters apart from numeric or decimal points in the price fields on Add Vehicle screen	Do not allow the characters to be entered	As expected	
Try to add a vehicle each time by leaving each one of the required fields blank on Add Vehicle screen	Display error message indicating some required fields are left blank and do not allow vehicle to be added	As expected	
Enter valid data into the fields and click Add Vehicle button	Add the vehicle details to the database and display in vehicle stock table	As expected	
Click View Vehicle Details button on stock list screen without selecting a vehicle from the table first	Display error message indicating vehicle from table must be selected and do not go to vehicle's details screen	As expected	
Click View Vehicle Details button on stock list screen by selecting a vehicle from the table first	Go to screen where all the vehicles details are displayed as stored in database	As expected but data for Colour field was empty despite its entry in database	Required variable that holds colour data was not assigned to colour retrieval from database
Click Edit Vehicle button on the view vehicle screen for a particular vehicle	Go to screen where user can edit the details, with the details already filled in the fields	As expected	
Change the details in all fields on the edit vehicle screen and click the Update Vehicle button	Go back to the view vehicle's details screen and see all details have been changed	As expected except for the 'Selling Price' field not being updated	The line of code to update the data entered in this field into the database was missing
Click the Delete Vehicle button when at the view vehicle's details screen for a particular vehicle	Display message box asking user to confirm whether they are sure they would like to delete the vehicle	As expected	
Click 'No' when asked when deleting vehicle	Do not delete the vehicle	As expected	
Click 'Yes' when asked when deleting a vehicle	Delete vehicle from database and go to back to stock screen	As expected	

Unit Testing for Implementing Process Sale of Vehicle

TEST	EXPECTED RESULT	ACTUAL RESULT	COMMENTS
Enter any characters apart from numeric or decimal points in the price fields on Sell Vehicle screen	Do not allow the characters to be entered	As expected	
Try to click the Process Sale button each time by leaving each one of the required fields blank on Sell Vehicle screen	Display error message indicating some required fields are left blank and do not allow vehicle to be sold	As expected	
Try to sell a vehicle with part-exchange selected but its fields left blank	Display error message indicating some required fields are left blank and do not allow vehicle to be sold	Processing of sale was allowed with SQL exception error thrown	No check was made on the part-exchange checkbox to ensure if it is selected, then the fields are filled in
Enter valid data into the fields and click the Process Sale button	Display confirmation box asking user to confirm the processing of the sale	As expected	
Click No when the confirmation box is displayed when trying to sell a vehicle	Do nothing and leave data previously entered in the fields to remain	As expected	
Click Yes when the confirmation box is displayed when trying to sell a vehicle	Add the data entered in the fields into the database, and go to previous sales screen with sale just processed displayed in table	As expected	
Open html file where sale details were written to	Data that was entered when selling the vehicle are shown in the html file	Data for 'Amount to Pay', 'Amount Paid', and 'Amount Due' were missing	Code to write these to the html file was missing
Click View Sale Details button on the previous sales screen without selecting a sale from the table first	Display error message indicating a sale from the table must be selected, and do not go to sale's details screen	As expected	
Click View Sale Details button on previous sales screen by selecting a sale from the table first	Go to screen where all the sale details are displayed as stored in database	As expected but data in customer Phone and Email fields were empty despite their entry in database	The phone and email columns in SQL query to retrieve data from customer table were missing
Click Edit Sale button on the view sale screen for a particular sale	Go to screen where user can edit the details, with the details already filled in the fields	As expected	
Change the details in all fields on the edit sale screen	Go back to the view sale's details screen and see all	As expected	

and click the Update Sale button	details have been changed		
Click the Delete Sale button when at the view sale's details screen for a particular sale	Display message box asking user to confirm whether they are sure they would like to delete the sale	As expected	
Click 'No' when asked when deleting sale	Do not delete the sale	As expected	
Click 'Yes' when asked when deleting a sale	Delete sale from database and go to back to previous sales screen	As expected	

Unit Testing for Implementing Schedule Vehicle for Repair or Service

TEST	EXPECTED RESULT	ACTUAL RESULT	COMMENTS
Select a date from the combo boxes and try to add new appointment without checking its availability first	Display error message indicating the date selected must be checked first for the number of appointments on that date	As expected	
Select a date from the combo boxes and click the Check Availability button	Display all appointments scheduled for that date in the table	As expected	
Click New Appointment button after selecting and checking the availability of a specific date	Go to add new appointment screen	As expected	
Enter non-numeric characters in 'Engine Size' fields on add appointment screen	Do not allow the non-numeric characters to be entered	As expected	
Try to add an appointment each time by leaving each one of the required fields blank	Display error message indicating some required fields are left blank and do not allow appointment to be added	As expected	
Enter valid data into the fields and click Add Appointment button	Add the appointment details to the database and display in appointments table	As expected	
Add six appointments on a particular date and then try to add a seventh to try and go over the maximum number currently allowed	Display error message indicating maximum allowed appointments on selected date is already reached	As expected	
Click View Appointment button on the scheduling screen without selecting an appointment from table first	Display error message indicating an appointment from the table must be selected first	Went to screen where all the appointment details are displayed as stored in database	Check for whether an appointment from the table was first selected was not implemented

Click View Appointment button on the scheduling screen by selecting an appointment from the table first	Go to screen where all the appointment details are displayed as stored in the database	As expected	
Click Edit Appointment button on the view appointment screen for a particular scheduled vehicle	Go to screen where user can edit the details, with the details already filled in the fields	As expected	
Change 'Date Scheduled' to a date where the maximum appointments allowed has already been reached	Display error message indicating maximum allowed appointments on selected date is already reached	The update of appointment was allowed on date with maximum allowed appointments reached	Functionality to check the maximum appointments allowed was not implemented
Change the details in all fields on the edit appointment screen and click Update Appointment button	Go back to the view appointment's details screen and see all details have been changed	As expected	
Click the Cancel/Delete button when at the view appointment's details screen for a scheduled vehicle	Display message box asking user to confirm whether they are sure they would like to cancel/delete the appointment	As expected	
Click 'No' when asked when deleting appointment	Do not delete the appointment	As expected	
Click 'Yes' when asked when deleting a appointment	Delete appointment from database and go to back to main scheduling screen	As expected	

User Acceptance Testing Feedback

After the assigned functionalities of the system for the first iteration were implemented, they were thoroughly tested by the employees of Steve Graves Motors. The following points are a summary of the feedback comments they gave during the test.

General Comments:

- The light blue used as the background for every screen sets a good interface standard for the whole system. The light soft yellow colour used for the background of the tabs was also nice, and the combination of the yellow tabs and blue background go well together.
- Titles are of a good size and font, and buttons are clearly defined in what they allow the user to do. Navigation is quite good as not many screens need to be navigated through.
- Drop down lists for some fields when entering data is useful as it does not require user to always have to enter common data already stored in database.

- The method of adding a vehicle to the stock is good, as data does not have to be entered in many places as required in the current system.
- System's ability to automatically fill in the details of a previous supplier in the fields when adding a new vehicle is good.
- The ease of processing the sale of a vehicle is also good. Data only has to be entered once in a single place, unlike the current system where two invoices have to be written as well as constantly updating the stock book.

Improvements and Modifications:

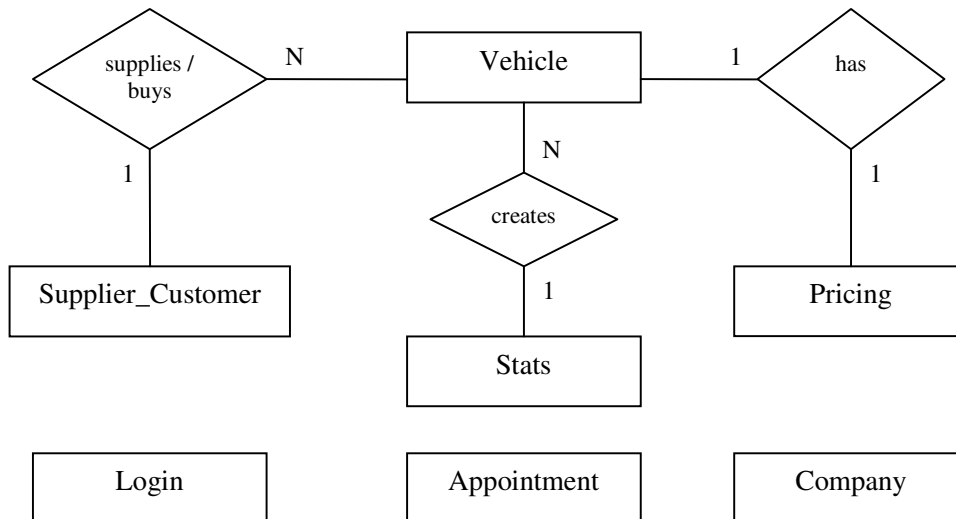
- Drop down list for the transmission field should be included rather than input text field as all vehicles will only be either manual or automatic.
- Include a field to store any additional information about a vehicle that may be required.
- A column for a vehicle's colour should be included in the vehicle stock list table.
- The price that a vehicle was bought for by the company should not be included on the screen to view a vehicle's full details as it needs to be hidden from a customer.
- Include an option to save a customer of a vehicle as a potential future supplier because some customers are other car sales companies and therefore could potentially be a supplier of a vehicle to this company in the future.
- Ensure the amount to pay, amount paid, and amount due fields are not left blank when processing the sale of a vehicle.
- When at a view or edit screen for a vehicle, sale, or appointment, give an indication as to exactly which item is being viewed or edited.

Additional Features:

- Ability to edit the details of a saved supplier.
- Ability to re-create a new sales invoice of a vehicle after it has been sold in case any details about the sale have been updated and customer requires a new receipt.
- Ability for the maximum number of appointments for a vehicle repair or service allowed to be scheduled on a single day to be changed.
- Ability to print out a vehicle's details currently for sale to give to a potentially interested customer.

APPENDIX G – Design of Second Iteration

Entity-Relationship Diagram of Final Database System



The rectangles in the diagram represent an entity, and a diamond represents a relationship. The Supplier_Customer table is related to the Vehicle table with a one to many relationship, this is because a supplier can supply many different vehicles to the company, and a customer can buy a vehicle. The Vehicle table is related to the Pricing table by a one to one relationship because for one vehicle there can only be one set of price figures associated with it. There can be many pricing values associated with a single vehicle before and after it is sold, therefore it was decided to keep the pricing figures for a vehicle in a separate table due to the Vehicle table already having to hold many attributes of a vehicle's details. The Stats table is related to the Vehicle table by a one to many relationship, this is because a number of different vehicles will help make up a single statistical record in the Stats database.

The Login and Company entities (tables) hold independent data from the rest of the database and therefore they do not require to be related to another entity. The Stats table is related to the Vehicle table with a *one to many* relationship, this is because many records about the make or body type of a vehicle in the Vehicle table can make up only one single statistical record for it in the Stats table. For example there may be a number of Ford cars in the Vehicle table, and they will all make up only one single record for the sales performance of Ford cars in the Stats table.

Functionality Design of Second Iteration

The ability to search for a particular type of vehicle from the stock list will require most of the fields from the vehicle tab used on previous screens as an input from the user. These will therefore be included on a screen where the user will be able to select their requirements. The search results will be displayed in the stock list table.

For the ability to search for a previous sale, the user will be able to enter the registration of the vehicle sold into an input box, and when found, the sale's details will appear in the previous sales table. From here the user will be able to view its additional details.

The functionality to automatically calculate the total VAT due will require a screen consisting of a table showing the details of vehicles with their VAT due value. A screen similar to the ones previously created that contain a table will be used as the layout. However the screen will have to be slightly modified to include a function to allow the dates of calculation required to be changed, and a field to show the total VAT due.

For calculating and comparing the total profit or loss made by the business, the screen for this was also based around a main table to show the information. A set of boxes to select dates the user wishes to view the sales figures for will also have to be included.

Invoicing a vehicle that has been repaired or serviced will simply include another set of tabs in addition to the current ones used for scheduling a vehicle. When the user goes to edit the appointment details, a new set of tabs will be provided to allow the user to enter the details of work carried out with prices into a number of text fields. A button will then allow the user to write the invoice details to a HTML file from where it can be printed off.

The login feature required as part of the new system is designed so that each time the system will be run, a small box will appear asking the user to enter the username and password that has been set. The system would then check the details with the data in the Login database and if the credentials match, then the main menu of the system will be displayed. In addition to this, a function will be required in the system for allowing the ability to change the password set on the system.

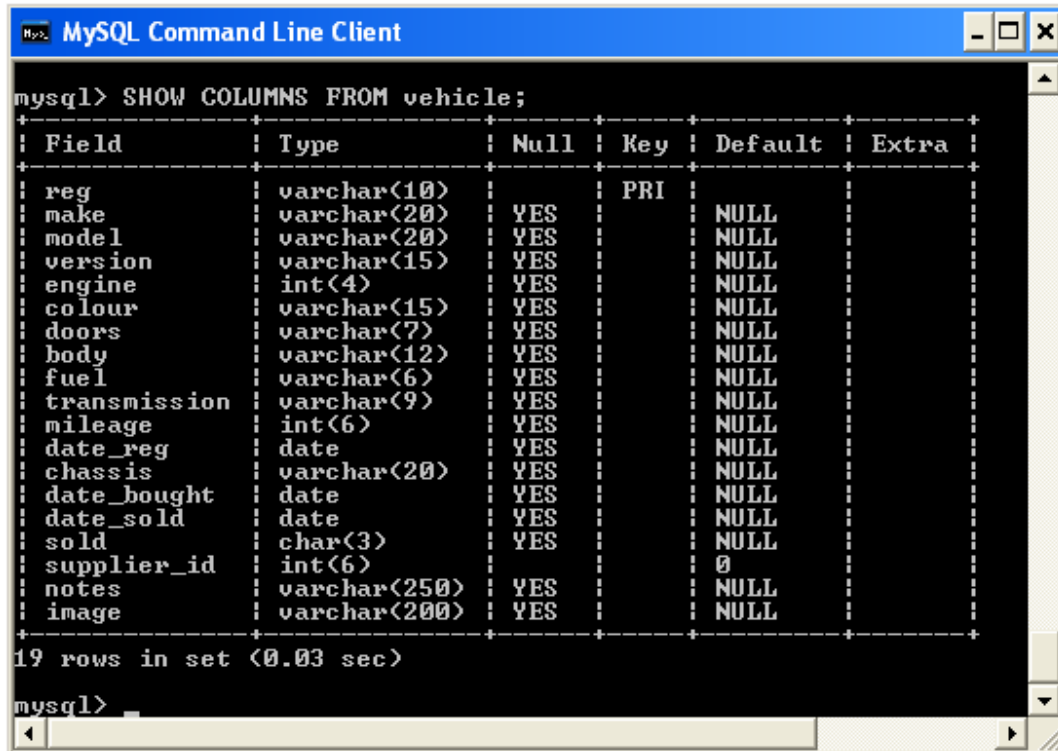
For the system's ability to view and compare the sales performance of vehicles by make or body style, the screen for this will be similar to when viewing the sales performance for the business. It will consist of a table showing a summary of the how many vehicles of a particular type have been sold, and its average profit. A feature to select whether the sales performance of vehicle makes or body styles will also be included.

To add an image of a vehicle that has been added to the stock, there will be a feature on the screens to add and edit a vehicle whereby when the user clicks a button, and a small window will appear allowing them to choose the image file of the vehicle. Once this is selected, it will be added onto the screen in the Image tab.

Personalising a company's details on the system will require the user going to a screen which will consist on a number of text fields already populated with the company's current address and contact details. Then the user will be able to make their changes in these boxes and update the details by clicking the update button.

Appendix H – Final Implementation of System

Vehicle Table



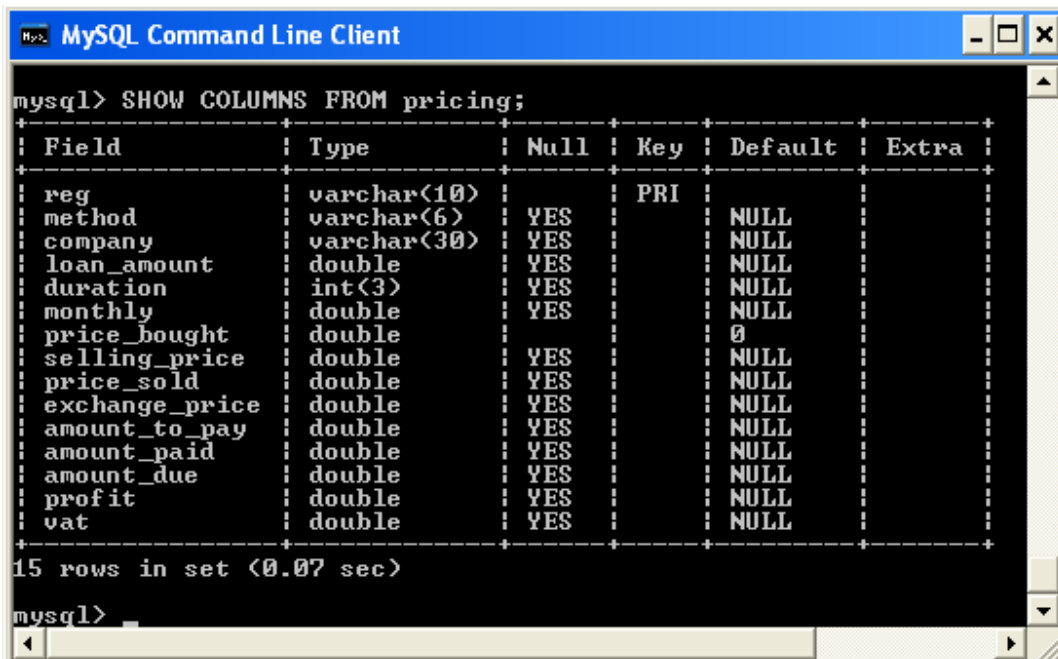
mysql> SHOW COLUMNS FROM vehicle;

Field	Type	Null	Key	Default	Extra
reg	varchar(10)		PRI		
make	varchar(20)	YES		NULL	
model	varchar(20)	YES		NULL	
version	varchar(15)	YES		NULL	
engine	int(4)	YES		NULL	
colour	varchar(15)	YES		NULL	
doors	varchar(7)	YES		NULL	
body	varchar(12)	YES		NULL	
fuel	varchar(6)	YES		NULL	
transmission	varchar(9)	YES		NULL	
mileage	int(6)	YES		NULL	
date_reg	date	YES		NULL	
chassis	varchar(20)	YES		NULL	
date_bought	date	YES		NULL	
date_sold	date	YES		NULL	
sold	char(3)	YES		NULL	
supplier_id	int(6)			0	
notes	varchar(250)	YES		NULL	
image	varchar(200)	YES		NULL	

19 rows in set (0.03 sec)

mysql>

Pricing Table



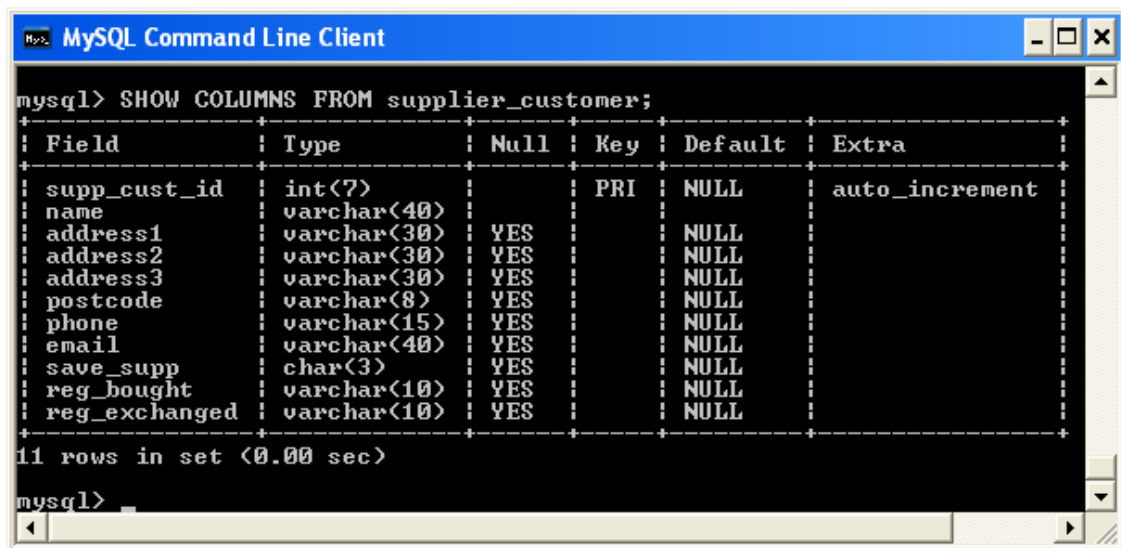
mysql> SHOW COLUMNS FROM pricing;

Field	Type	Null	Key	Default	Extra
reg	varchar(10)		PRI		
method	varchar(6)	YES		NULL	
company	varchar(30)	YES		NULL	
loan_amount	double	YES		NULL	
duration	int(3)	YES		NULL	
monthly	double	YES		NULL	
price_bought	double			0	
selling_price	double	YES		NULL	
price_sold	double	YES		NULL	
exchange_price	double	YES		NULL	
amount_to_pay	double	YES		NULL	
amount_paid	double	YES		NULL	
amount_due	double	YES		NULL	
profit	double	YES		NULL	
vat	double	YES		NULL	

15 rows in set (0.07 sec)

mysql>

Supplier_Customer Table



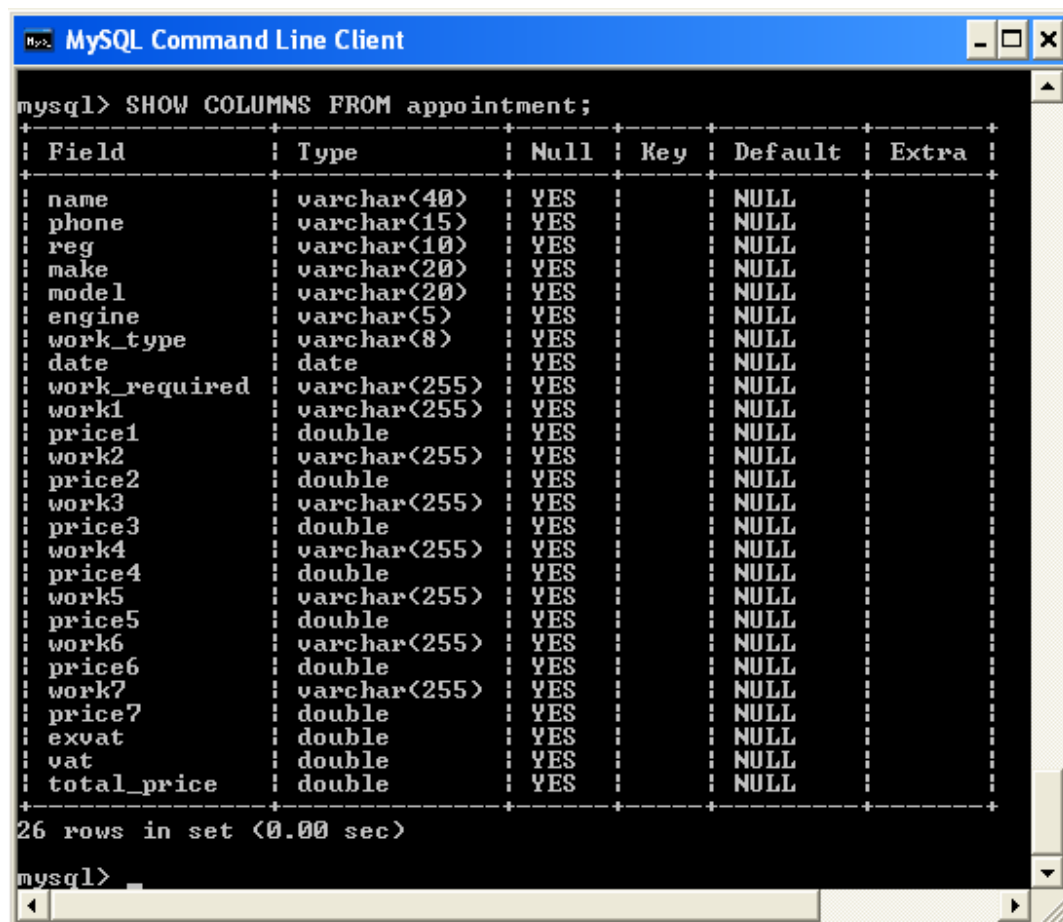
mysql> SHOW COLUMNS FROM supplier_customer;

Field	Type	Null	Key	Default	Extra
supp_cust_id	int(7)		PRI	NULL	auto_increment
name	varchar(40)				
address1	varchar(30)	YES		NULL	
address2	varchar(30)	YES		NULL	
address3	varchar(30)	YES		NULL	
postcode	varchar(8)	YES		NULL	
phone	varchar(15)	YES		NULL	
email	varchar(40)	YES		NULL	
save_supp	char(3)	YES		NULL	
reg_bought	varchar(10)	YES		NULL	
reg_exchanged	varchar(10)	YES		NULL	

11 rows in set (0.00 sec)

mysql>

Appointment Table



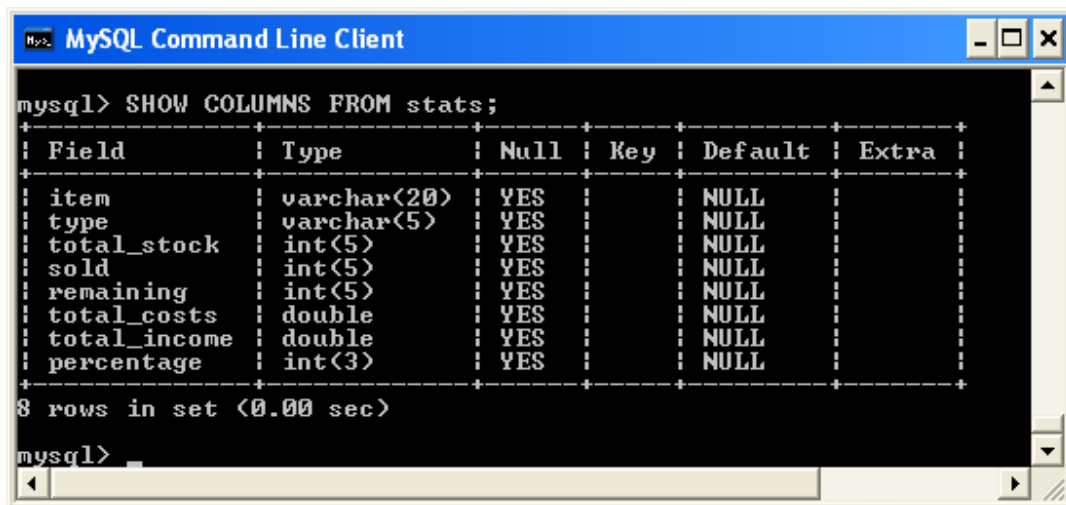
mysql> SHOW COLUMNS FROM appointment;

Field	Type	Null	Key	Default	Extra
name	varchar(40)	YES		NULL	
phone	varchar(15)	YES		NULL	
reg	varchar(10)	YES		NULL	
make	varchar(20)	YES		NULL	
model	varchar(20)	YES		NULL	
engine	varchar(5)	YES		NULL	
work_type	varchar(8)	YES		NULL	
date	date	YES		NULL	
work_required	varchar(255)	YES		NULL	
work1	varchar(255)	YES		NULL	
price1	double	YES		NULL	
work2	varchar(255)	YES		NULL	
price2	double	YES		NULL	
work3	varchar(255)	YES		NULL	
price3	double	YES		NULL	
work4	varchar(255)	YES		NULL	
price4	double	YES		NULL	
work5	varchar(255)	YES		NULL	
price5	double	YES		NULL	
work6	varchar(255)	YES		NULL	
price6	double	YES		NULL	
work7	varchar(255)	YES		NULL	
price7	double	YES		NULL	
exvat	double	YES		NULL	
vat	double	YES		NULL	
total_price	double	YES		NULL	

26 rows in set (0.00 sec)

mysql>

Stats Table



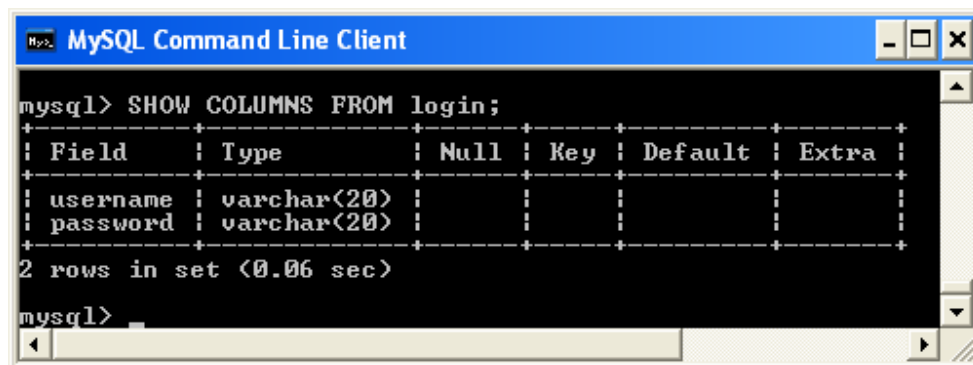
mysql> SHOW COLUMNS FROM stats;

Field	Type	Null	Key	Default	Extra
item	varchar(20)	YES		NULL	
type	varchar(5)	YES		NULL	
total_stock	int(5)	YES		NULL	
sold	int(5)	YES		NULL	
remaining	int(5)	YES		NULL	
total_costs	double	YES		NULL	
total_income	double	YES		NULL	
percentage	int(3)	YES		NULL	

8 rows in set (0.00 sec)

mysql>

Login Table



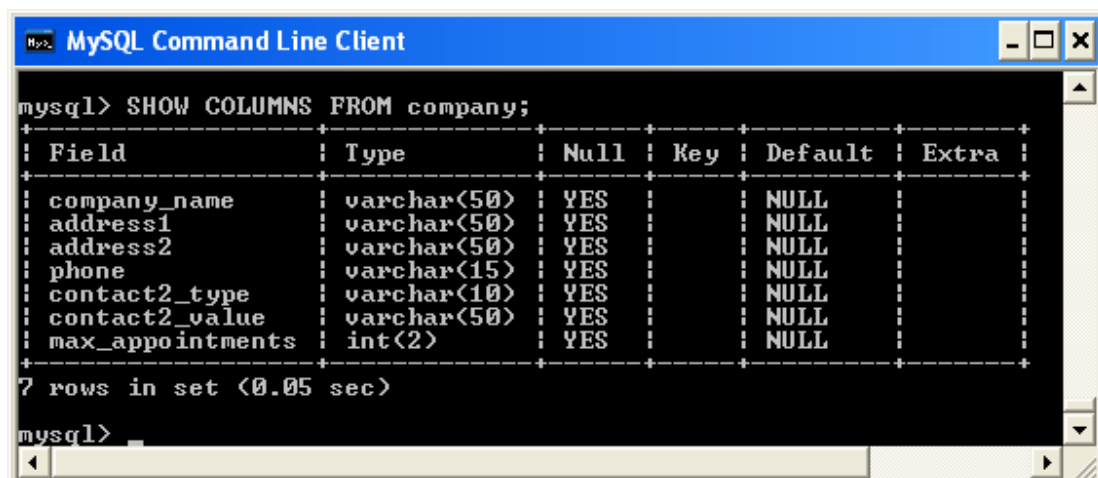
mysql> SHOW COLUMNS FROM login;

Field	Type	Null	Key	Default	Extra
username	varchar(20)				
password	varchar(20)				

2 rows in set (0.06 sec)

mysql>

Company Table



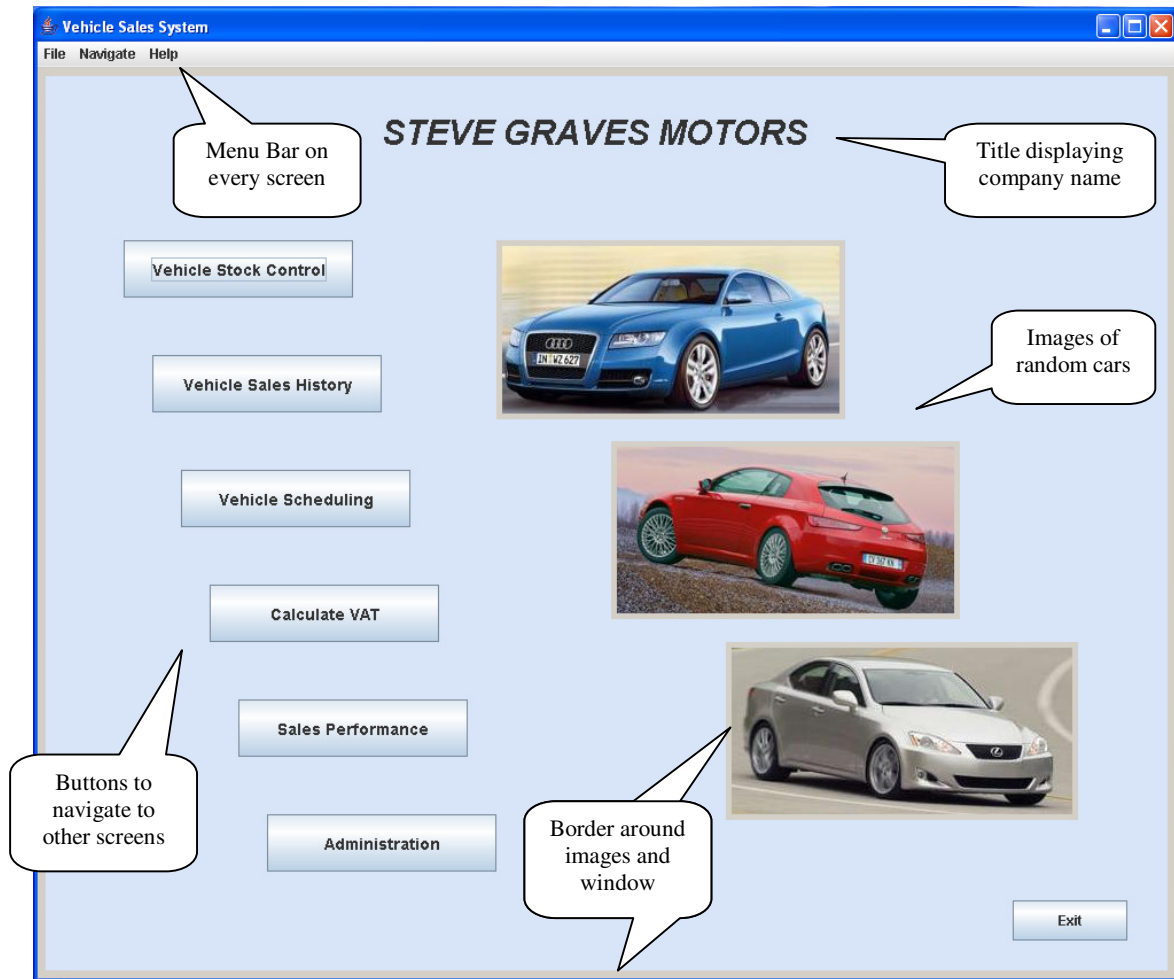
mysql> SHOW COLUMNS FROM company;

Field	Type	Null	Key	Default	Extra
company_name	varchar(50)	YES		NULL	
address1	varchar(50)	YES		NULL	
address2	varchar(50)	YES		NULL	
phone	varchar(15)	YES		NULL	
contact2_type	varchar(10)	YES		NULL	
contact2_value	varchar(50)	YES		NULL	
max_appointments	int(2)	YES		NULL	

7 rows in set (0.05 sec)

mysql>

Implementation of Main Menu



The title was created using a *JLabel* component in Java which creates a label, the text added to it was the company name retrieved from the Company table in the database. The font was set to size 24, with bold and italic effects. The code for creating the title on the main menu screen consisted of the following, with a similar format used throughout the other parts of the system:

```
JLabel companyLabel = new JLabel(companyName);  
companyLabel.setFont(new Font("Dialog", Font.BOLD | Font.ITALIC, 30));
```

Borders were also set around the images and the main frame of the window. This was to give a better look to the system. The code to add a border around a component consisted of:

```
menuPanel.setBorder(BorderFactory.createLineBorder(new Color(212, 208, 200), 7));  
imageLabel.setBorder(BorderFactory.createLineBorder(new Color(212, 208, 200), 5));
```

Manage Vehicle Stock Control – Add Vehicle

The screenshot shows a Java Swing window titled "Vehicle Sales System" with a menu bar (File, Navigate, Help). The main content area is titled "ADD VEHICLE" and features a tabbed interface with "Vehicle", "Image", and "Supplier" tabs. The "Vehicle" tab is active, displaying a form with the following fields:

- *Registration Number: YK05R8T
- Make: Volkswagen (dropdown)
- Model: (dropdown) with "Golf" as a visible option
- Version: GTI
- Engine Size: 2000 cc
- Colour: Silver
- Number of Doors: 5 Doors (dropdown)
- Body Style: Hatchback (dropdown)
- Fuel Type: Petrol (dropdown)
- Transmission: Manual (dropdown)
- Mileage: 25000
- *Date First Registered: 2005-06-22 (format: yyyy-mm-dd)
- Chassis Number: JD83HD74HF148DJH47

At the bottom of the form are three buttons: "Add Vehicle", "Edit Supplier", and "Cancel". A small asterisk indicates that fields marked with * are required.

Callouts in the image:

- "Tabs" points to the "Vehicle", "Image", and "Supplier" tabs.
- "Field labels created using JLabel" points to the labels like "Registration Number:", "Make:", etc.
- "Input text fields created using JTextField" points to text boxes like "Registration Number", "Version", "Engine Size", "Colour", "Mileage", and "Chassis Number".
- "Drop down combo boxes created using JComboBox" points to dropdown menus like "Make:", "Model:", "Number of Doors:", "Body Style:", "Fuel Type:", and "Transmission:".

The fields for inputting data into the database consist of text fields and drop down combo boxes. Text fields were created with *JTextField* components in Java, and combo boxes were created using *JComboBox* components. The labels at the side of each field were created using *JLabels*. The code to create labels, text fields, and combo boxes consisted of the following format:

```
JLabel fieldLabel = new JLabel("Label Name"); //create field label
JTextField field = new JTextField(); // create text field
JComboBox box = new JComboBox(); // create combo box
box.addItem("Item Name"); // add items to combo box
```

This code to create labels, text fields, and combo boxes was used throughout various parts of the system where they were required to be created. The variable names were replaced with the relevant variable names required for the class that carries out a certain functionality.

In the supplier tab, if the new vehicle being added is supplied by a previous supplier, the user can select the name from the Previous Supplier combo box and all the other fields will be filled in automatically. If it is a new supplier, the user can select 'None' from the combo box and enter in the new supplier's details in the fields. If they wish to save this supplier as a potential future supplier so that next time their name would appear in the Previous Supplier combo box, then they can do so by ticking the 'Save as future supplier' checkbox. A snippet of the code for automatically filling in the Name field if a previous supplier is selected consists of the following:

```
if (supplierBox.getSelectedItem().equals("None")) {
    nameField.setText("");
    nameField.setEditable(true);
} else {
    nameField.setText(results.getString("name"));
    nameField.setEditable(false);
}
```

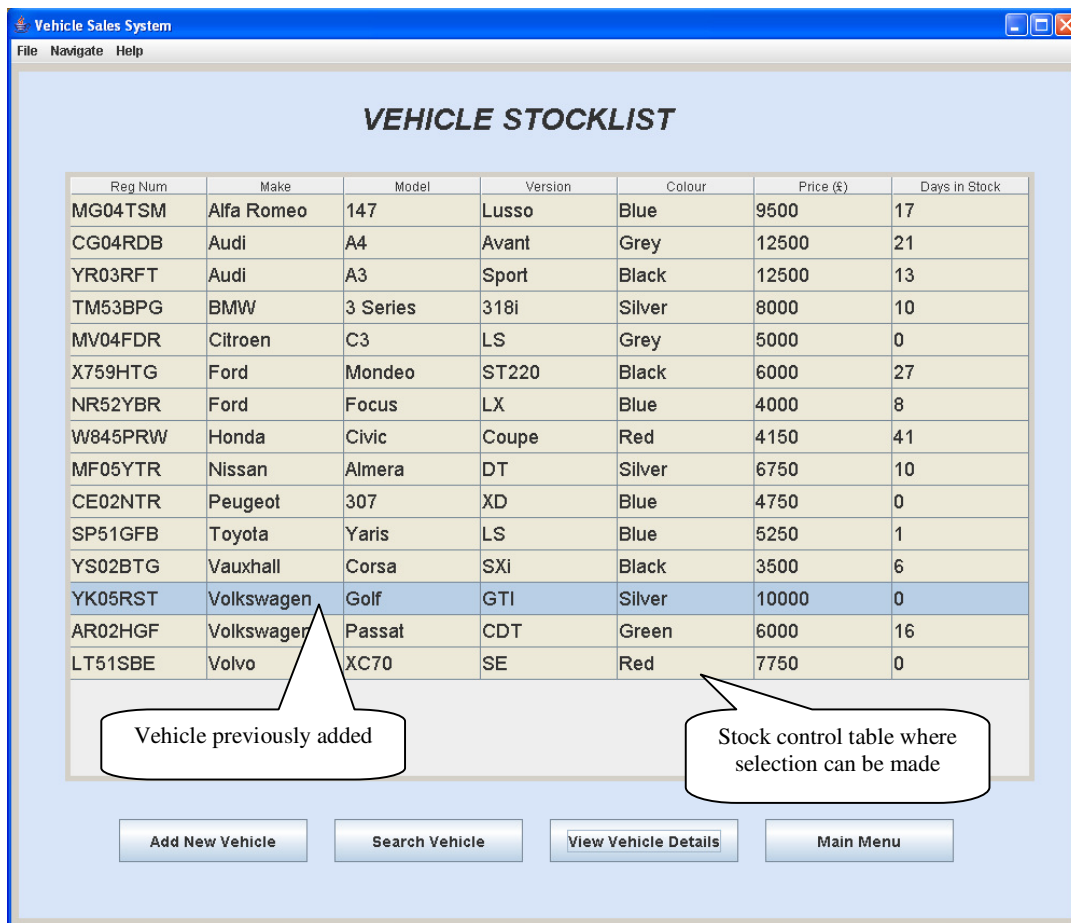
If any of the required fields (marked by an *) are left blank and the user tries to click the Add Vehicle button on the screen, the following error message is displayed.



The code for this feature consists of the following, with a similar format used for other screens:

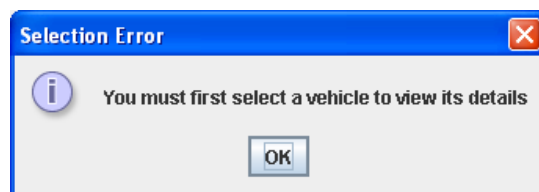
```
if (regField.getText().equals("") || dateRegField.getText().equals("") ||
    nameField.getText().equals("") || priceBoughtField.getText().equals("")) {
    JOptionPane.showMessageDialog(null, "Some of the required fields are
incomplete", "Information Entry Error", JOptionPane.INFORMATION_MESSAGE);
}
```

Manage Vehicle Stock Control – Main Stock Control Screen

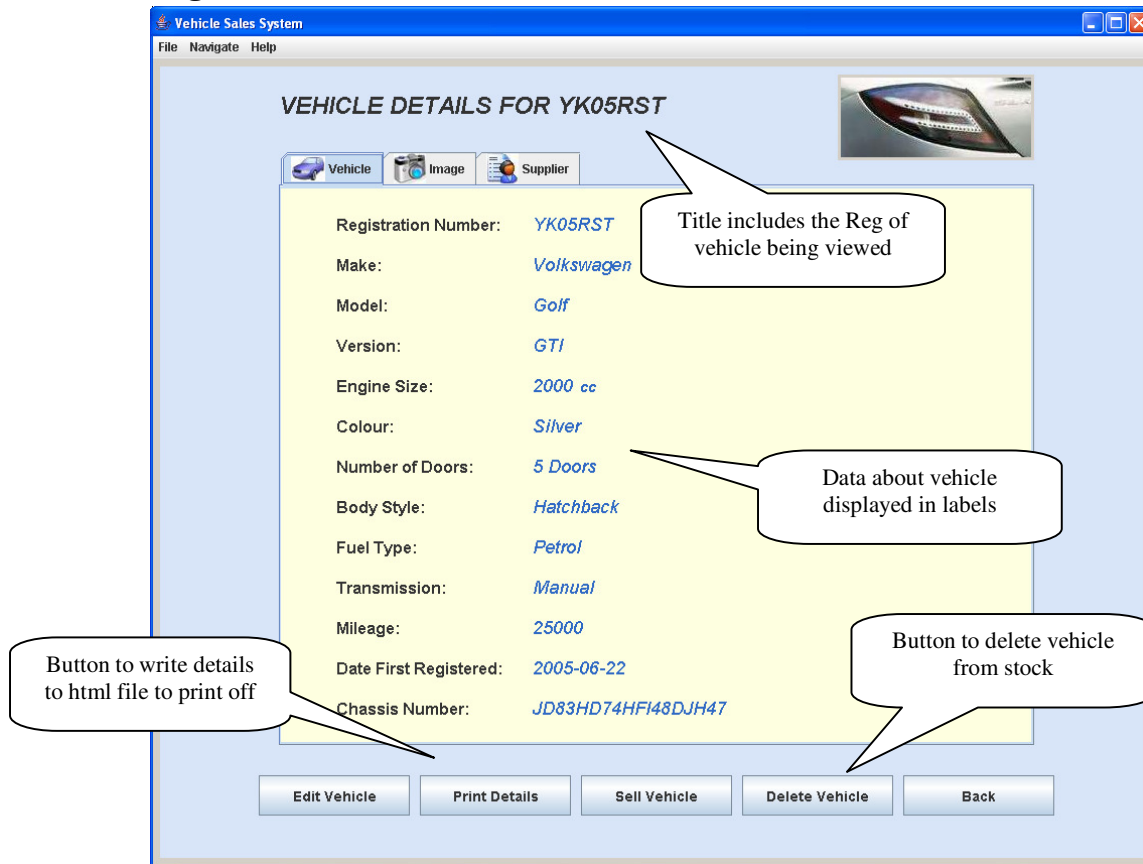


When a new vehicle is added to the stock, it is displayed on the main stock control screen in the table. The table is created by using a *JTable* component in Java. From here the user can select a vehicle they wish to view, and then select the View Vehicle Details button. If the user tries to click this button without selecting a vehicle first, the error message below is displayed. The code for this is:

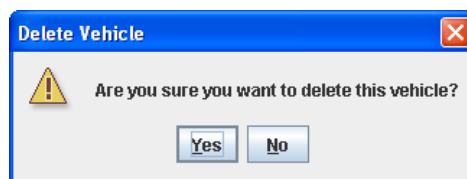
```
if (stockTable.getSelectedRow() != -1) {
    String selection = (String) list.get(stockTable.getSelectedRow());
    viewVehicle = new ViewVehicle(gui, selection, query);
    gui.switchPanels(viewVehicle.returnViewPanel());
} else {
    JOptionPane.showMessageDialog(null, "You must first select a vehicle to view its details", "Selection Error", JOptionPane.INFORMATION_MESSAGE);
}
```



Manage Vehicle Stock Control – View Vehicle



On the view vehicle screen, the data about the vehicle retrieved from the database is displayed in labels. This is because it gives a better presentation on the screen rather than using text fields populated the data. These labels are set to a dark blue font colour, with an italic style. When the Delete Vehicle button is clicked, the following message box is displayed to confirm the deletion.



When the Edit Vehicle button is clicked, the same screen used for adding a vehicle is displayed but with the vehicle's details populated in all the fields. From here the user will be able to edit the required fields and click the Update Vehicle button which will update the details in the database.

```
editVehicle = new AddEditVehicle(gui, reg);
gui.switchPanels(editVehicle.returnAddPanel());
editVehicle.editVehicle(reg, make, model, version, engine, colour, doors,
body, fuel, transmission, mileage, dateReg, chassis, date, supplierID,
name, address1, address2, address3, postcode, phone, priceBought,
sellingPrice, notesText, image);
```

Printout of Vehicle Details

When the Print Details button is click when viewing the details of a vehicle, a HTML file is created which can then be printed off. A printout of the details of a vehicle is shown below.

STEVE GRAVES MOTORS

Unit 1A Huddersfield Road, Birstall, WF17 9AY
TEL: 01924 475382, Fax: 01924 475282

VEHICLE QUOTATION

Reg No:	YK05RST
Make:	Volkswagen
Model:	Golf GTI
Engine Size:	2000 cc
Colour:	Silver
No. of Doors:	5 Doors
Body Style:	Hatchback
Fuel:	Petrol
Transmission:	Manual
Mileage:	25000
Date Registered:	2005-06-22
Additional Notes:	Full service history, alloy wheels, air conditioning, leather seats, 12 months road tax, 6 months MOT
Price:	£10000



Process Sale of Vehicle

Vehicle Sales System
File Navigate Help

VEHICLE SALE INVOICE

Vehicle Customer Part-Exchange Pricing

Title: Mr. [v]
First Name: Thomas
*Surname: Phillips
Address: 36 Hyde Road
Shipley
Bradford
Postcode: BD7 4FR
Phone: 01274 248751
Email: thomasphillips@email.com

☐ Save as future supplier if part-exchanged

* Required Field

Process Sale Cancel

Ability to save customer as potential future supplier

Text fields for entering customer's details

The Vehicle tab consists of the same fields used from the add vehicle screen, but with data about the vehicle already populated in the text fields. The checkbox on the screen allows the customer to be saved as a potential future supplier if ticked. If the box is not ticked, then the customer's details are entered in the Supplier-Customer table but with the save_supp field set to 'no', as a result the customer's details will not appear in the supplier combo box when adding a vehicle. An *ActionListener* was set on the checkbox so that each time it is ticked or un-ticked, the correct action will be taken to set or not to set the supplier as a potential future supplier. The code within the *ActionListener* method for the checkbox is of the following format:

```
private String save = "no";  
if (save == "no") {  
    save = "yes";  
} else {  
    save = "no";  
}
```

☐ Customer does not wish to part-exchange vehicle

*Registration Number: R235BTD

Make: Honda

Model: Civic

Version: XT

Engine Size: 1600 cc

Colour: Maroon

Number of Doors: 5 Doors

Body Style: Hatchback

Fuel Type: Petrol

Transmission: Automatic

Mileage: 85000

*Date First Registered: 1998-05-04 (yyyy-mm-dd)

Chassis Number: JF74N7FH57FHJ47

* Required Field

If customer not part-exchanging, tick this box.

Honda Civic already in database so make and model name can be selected from combo box

The Part-Exchange tab allows the user to enter the details of a vehicle that the customer wishes to part exchange when buying a vehicle. If they are not part exchanging a vehicle then the user must tick the checkbox at the top indicating that a part exchange is not to be carried out. As a result the system will not attempt to enter in a new record in the vehicle table for adding a new vehicle. In the pricing tab, the price fields had to be validated so that only numerical and decimal point characters can be entered in the field. A method was set upon the fields where a price is required to be entered. The code within the method to validate the price fields consists of:

```
textField.addKeyListener(new KeyAdapter() {
    public void keyTyped(KeyEvent e) {
        char c = e.getKeyChar();
        if (!((Character.isDigit(c) ||
            (c == KeyEvent.VK_BACK_SPACE) ||
            (c == KeyEvent.VK_PERIOD) ||
            (c == KeyEvent.VK_DELETE)))) {
            e.consume();
        }
    }
});
```

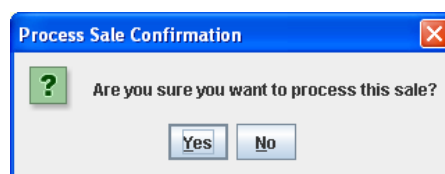
In the above code, *textField* is a parameter of the method so that for any text field that needs this validation, the name of that text field just has to be passed as a parameter to this method.

The screenshot shows the 'Pricing' tab of a software interface. It contains the following fields and callouts:

- Date of Sale:** 11 April 2007. Callout: "Date derived and inserted automatically by system".
- Method of Payment:** Card (dropdown). Callout: "Combo box to choose the method of payment".
- Finance Company:** (text field).
- Amount of Loan:** £ (text field).
- Duration:** (text field) months.
- Payment per Month:** £ (text field).
- List Price:** £ 10000.
- *Final Price:** £ 9800.
- Part Exchange Value:** £ 2000.
- *Amount to Pay:** £ 7800.
- *Amount Paid:** £ 100.
- *Amount Due:** £ 7700.
- Callout for the last four fields: "Price fields".
- Callout for the last four fields: "If method of payment is 'Credit', then these fields will be set to enabled to allow data entry".
- Legend: * Required Field.

In the Pricing tab, if 'Credit' is selected as the Method of Payment, then the four fields below it will become active to allow data to be inserted into them. This is because these fields are only required to be filled in if the customer is purchasing the vehicle with monthly finance credit. With any other selection, these fields are set to be disabled. When the other fields are complete and the Process Sale button is click, the confirmation message below is displayed, created with the following code:

```
int choice = JOptionPane.showConfirmDialog(null, "Are you sure you
want to process this sale?", "Process Sale Confirmation",
JOptionPane.YES_NO_OPTION, JOptionPane.QUESTION_MESSAGE);
```



If 'Yes' is clicked in the Process Sale Confirmation box, the methods to add the relevant data into the database are called. This is implemented by the following code:

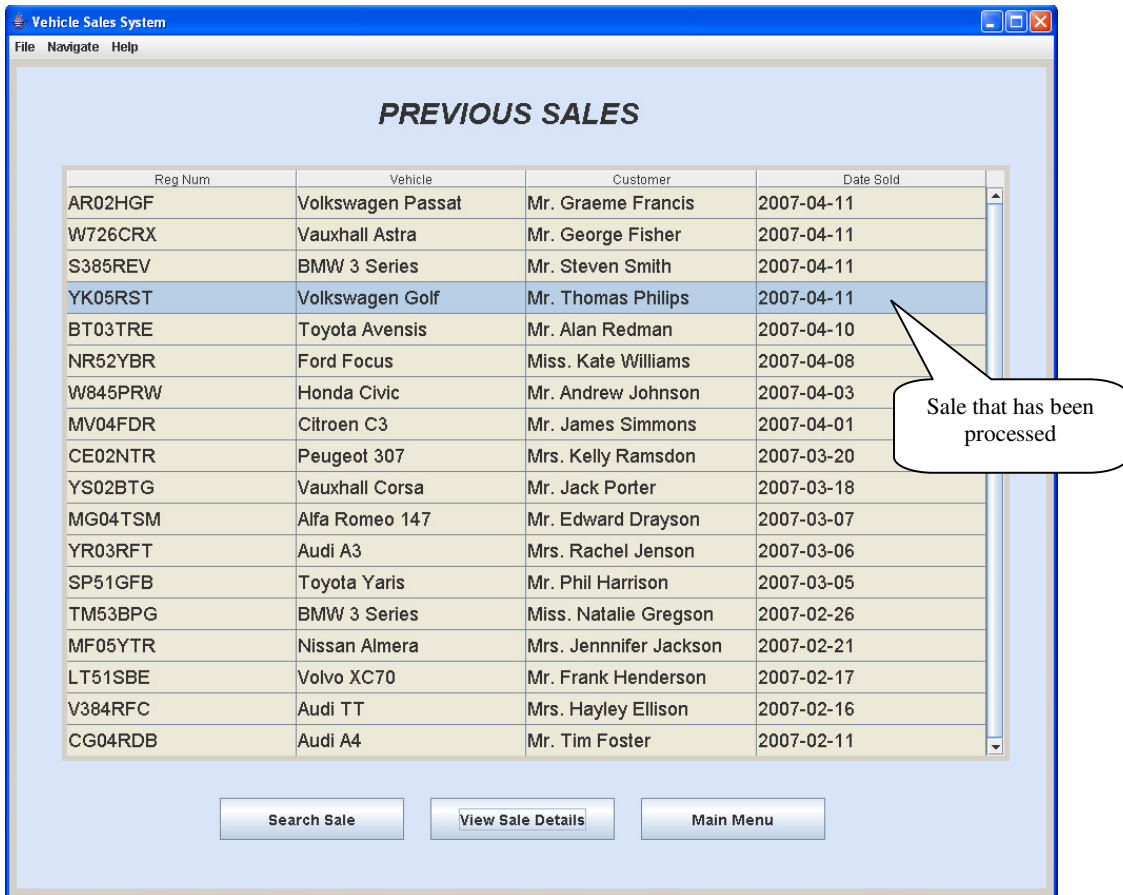
```
if (choice == JOptionPane.YES_OPTION) {
    updateVehicleSold();
    addCustomer();
    addPricing();
    getInvoice();
    if (exchange == "true") {
        addExchangeVehicle();
    }
}
```

Sales Invoice Printout

A printout from the system of the sales invoice for a vehicle that has been sold is shown below.

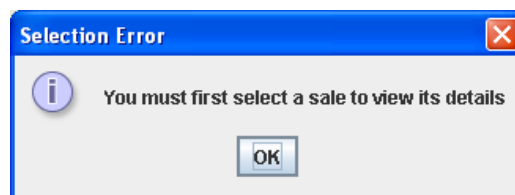
STEVE GRAVES MOTORS	
Unit 1A Huddersfield Road, Birstall, WF17 9AY TEL: 01924 475382, Fax: 01924 475282	
VEHICLE SALES INVOICE	
CUSTOMER	VEHICLE SOLD
Name: Mr. Thomas Philips	Reg No: YK05RST
Address: 36 Hyde Road Shipley Bradford	Make: Volkswagen
Postcode: BD7 4FR	Model & Version: Golf GTI
Telephone: 01274 248751	Engine Size: 2000 cc
Date Sold: 11 April 2007	Colour: Silver
	Mileage: 25000
	Date First Registered: 2005-06-22
	Chassis: JD83HD74HFI48DJH47
PART EXCHANGE VEHICLE	PRICING
I/we certify this vehicle conforms to the following description, is my/our own property, and free from any legal encumbrances.	
Reg No: R235BTD	Vehicle Price: £9800
Make: Honda	Part Exchange: £2000
Model & Version: Civic XT	Amount to Pay: £7800
Engine Size: 1600 cc	Amount Paid: £100
Colour: Maroon	Amount Due: £7700
Mileage: 85000	
Date First Registered: 1998-05-04	
Chassis No: JF74N7FH57FHJ47	
Declaration by Customer: I/we declare that I/we am/are the buyer(s) of the vehicle sold described above I/we acknowledge receiving delivery of the vehicle sold described above Signature(s)..... Date.....	
Declaration by Seller: Input tax deduction has not been and will not be claimed by me in respect of the vehicle sold Signature..... Date.....	

Process Sale of Vehicle – View Sale



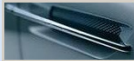
When the sale is processed it appears in the table in the Previous Sales screen. From here the user can select the sale they wish to view, and then click the View Sale Details button. If the user tries to click the View button without selecting a sale from the table first, the error message below is displayed. The code for this consists of:

```
if (salesTable.getSelectedRow() != -1) {  
    String selection = (String) list.get(salesTable.getSelectedRow());  
    ViewSale sale = new ViewSale(gui, selection);  
    gui.switchPanels(sale.returnViewPanel());  
} else {  
    JOptionPane.showMessageDialog(null, "You must first select a  
sale to view its details",  
    "Selection Error", JOptionPane.INFORMATION_MESSAGE);  
}
```



Vehicle Sales System
File Navigate Help

SALE DETAILS FOR YK05RST



Vehicle Sold Image Supplier Customer Part-Exchanged Pricing

Registration Number: *YK05RST*
 Make: *Volkswagen*
 Model: *Golf*
 Version: *GTI*
 Engine Size: *2000 cc*
 Colour: *Silver*
 Number of Doors: *5 Doors*
 Body Style: *Hatchback*
 Fuel Type: *Petrol*
 Transmission: *Manual*
 Mileage: *25000*
 Date First Registered: *2005-06-22*
 Chassis Number: *JD83HD74HFI48DJH47*

Edit Sale Create Invoice Delete Sale Back

Vehicle Sold Image Supplier Customer Part

Name: *Mr. Thomas Philips*
 Address: *36 Hyde Road*
Shipley
Bradford
 Postcode: *BD7 4FR*
 Phone: *01274 248751*
 Email: *thomasphilips@email.com*

Vehicle Sold Image Supplier Customer Part-Exchanged Pricing

Date of Sale: *2007-04-11*
 Method of Payment: *Card*
 Finance Company:
 Amount of Loan: *£ 0*
 Duration: *0 months*
 Payment per Month: *£ 0*
 List Price: *£ 10000*
 Price Sold: *£ 9800*
 Part Exchange Value: *£ 2000*
 Amount to Pay: *£ 7800*
 Amount Paid: *£ 100*
 Amount Due: *£ 7700*

The sale details can be viewed at the view sale details screen, which displays all the information about the sale stored in the database when the sale was processed. From here the user will be able to edit or delete the sale details, and re-create a new invoice with the current data. The code for the *ActionListener* set upon the Edit Sale button consists of:

```
editSale = new EditSale(gui, reg);
gui.switchPanels(editSale.returnEditSalePanel());
```

VEHICLE STOCKLIST

Reg Num	Make	Model	Version	Colour	Price (£)	Days in Stock
VT51RXT	Alfa Romeo	GT	Sport	Blue	7500	25
GF51TVB	Citroen	C5	CD	Grey	4000	43
FD02BDM	Ford	Fiesta	Zetec	Grey	3500	43
X759HTG	Ford	Mondeo	ST220	Black	6000	56
P468VBT	Ford	Mondeo	LX	Red	2500	61
R638BGT	Honda	Accord	SE	White	2250	33
R235BTD	Honda	Civic	XT	Maroon		1
T372RDE	Nissan	Micra	C	Red	1750	23
T238RVY	Peugeot	406	SW	White	2000	41

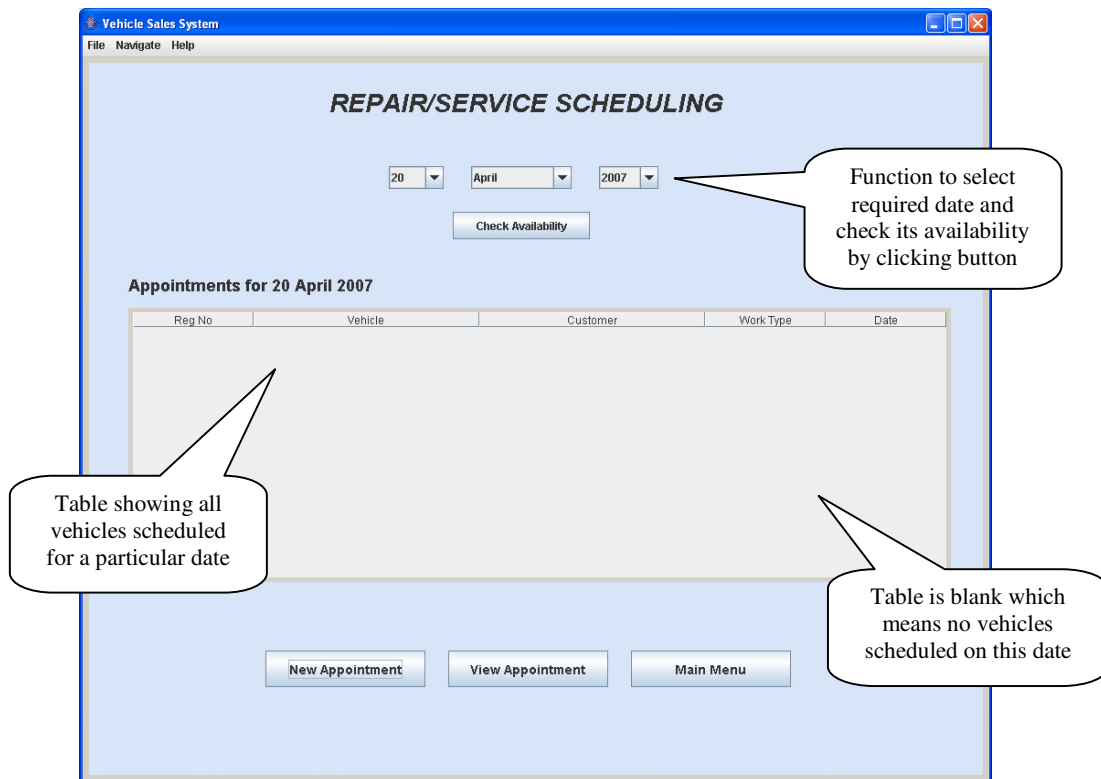
The vehicle that was part-exchanged during the previous sale now appears automatically in the vehicle stock list table after the sale has been processed. The selling price field is initially blank as this information about how much the vehicle they part-exchanged for will be sold for must be kept hidden from the customer. The user can however edit the vehicle's details and add a selling price.

Edit Supplier

When the Edit Supplier button is clicked on the add new vehicle screen, a new window appears allowing the user to select which supplier they wish to edit from the combo box. When the supplier is selected their details are filled in the fields, and the user can edit the details in the fields (right). When Update is clicked the details are updated in the database, this can be seen in the supplier tab when the edited supplier is selected from the combo box. The code that updates the supplier's details in the database is shown below:

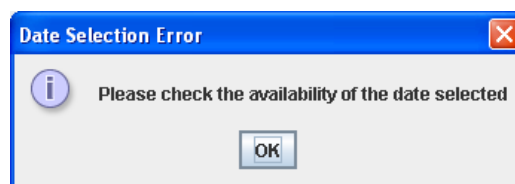
```
db = new DBConnection();
try {
    db.insertQuery("UPDATE supplier_customer SET " +
        "name = '" + nameField.getText() +
        "', address1 = '" + address1Field.getText() +
        "', address2 = '" + address2Field.getText() +
        "', address3 = '" + address3Field.getText() +
        "', postcode = '" + postcodeField.getText() +
        "', phone = '" + phoneField.getText() +
        "', email = '" + emailField.getText() +
        "' WHERE supp_cust_id = '" + previousSupplierID + "' ");
    db.closeDB();
}
```

Schedule Vehicle for Repair or Service – *Check Availability*



The main screen for when scheduling a vehicle for a repair or service consists of a table showing all the vehicles scheduled in the future. The user must check the availability of the date first to make sure it has not reached its maximum allowed for that day. This is done by selecting the required date from the combo boxes at the top of the screen and then clicking the Check Availability button below it. The table will then be updated with the vehicles scheduled for that date. If there are slots available, the user will be allowed to click the New Appointment button. However if the date is fully booked or the availability has not been checked, the error message below is displayed. The code to check that the availability of the selected date is valid consists of the following:

```
String selectedDate = dayBox.getSelectedItem() + " " +
monthBox.getSelectedItem() + " " + yearBox.getSelectedItem();
if (selectedDate.equals(dateFormatted) && numAppointments.size() <
getMaxApps()) {
AddEditAppointment addAppointment = new AddEditAppointment(gui, dateWanted,
dateFormatted);
gui.switchPanels(addAppointment.returnAddAppPanel());
} else {
JOptionPane.showMessageDialog(null, "Please check the availability of the date
selected", "Date Selection Error", JOptionPane.INFORMATION_MESSAGE);
```



Schedule Vehicle – Add New Appointment

Vehicle Sales System
File Navigate Help

NEW APPOINTMENT FOR 20 April 2007

Customer & Vehicle Work Required

*Customer Name: William Clarkson

Customer Phone: 07815032729

*Vehicle Reg No: VR51DTN

Vehicle Make: Ford

Vehicle Model: Focus

Engine Size: 1600 cc

* Required Field

Add Appointment Cancel

The title on the screen was created using a *JLabel* and includes the date of when the appointment is to be scheduled, this makes good usability of the system. The input in the fields in the Customer & Vehicle tab is via normal text fields. In the Work Required tab, the user can enter details of what is required to be carried out. The date field is again filled in automatically by the system for the date to be scheduled. The Work Requirements field makes use of a text area *JTextArea* where large amounts of information can be entered. The code in the method that adds the data entered in the fields into the database is shown on the right:

Customer & Vehicle Work Required

Type of Work: Service

Date Scheduled: 20 April 2007

Work Requirements: Check brakes and tyres, top up all oils, check exhaust, replace rear brake light, fix door hinge

```
db.insertQuery("INSERT INTO appointment VALUES ('" +
    nameField.getText() + "','" +
    contactField.getText() + "','" +
    regField.getText() + "','" +
    makeField.getText() + "','" +
    modelField.getText() + "','" +
    engineField.getText() + "','" +
    typeField.getSelectedItem() + "','" +
    dateWanted + "','" +
    requirementsField.getText() + "','" +
    "'','0','0','0','0','0','0','0','0','0','0','0','0','0','0','0','0' +
    '0','0','0','0' + '")");
db.closeDB();
```

Schedule Vehicle – View Appointment

20 April 2007

Check Availability

Appointment now appears in table

Appointments for 20 April 2007

Reg No	Vehicle	Customer	Work Type	Date
VR51DTN	Ford Focus	William Clarkson	Service	2007-04-20

Vehicle Sales System

File Navigate Help

APPOINTMENT DETAILS FOR VR51DTN

Vehicle & Customer Work Required Work Performed Pricing

Customer Name: William Clarkson

Customer Phone: 07815032729

Vehicle Reg No: VR51DTN

Vehicle Make: Ford

Vehicle Model: Focus

Engine Size: 1600 cc

Button to allow invoicing a vehicle scheduled with specific work carried out

Button to write invoice details to HTML file where it can be printed off

Tabs appear blank at first. But data can be added when appointment edited

Edit Appointment Create Invoice Cancel / Delete Back

After the new appointment has been added, it appears in the schedule table from where the user can select it and click the View Appointment button. The screen containing the appointment is displayed with its details from the database displayed in a number of labels (*JLabels*). The Work Performed and Pricing tabs appear when a new appointment is added, however they are currently blank. When the user edits the appointment they can enter in the details in these tabs which effectively allows them to invoice the work carried out on a vehicle.

Search Vehicle

SEARCH VEHICLE

Registration No:

Make:

Model:

Number of Doors:

Body Style:

Colour:

Fuel:

Transmission:

Maximum Age:

Maximum Mileage:

Price Range: Min: £ Max: £

Combo boxes to allow easier selection choice for requirements

Ability to search within specified price bracket makes for good usability of search feature

The search vehicle feature brings up a new window (shown above) mainly consisting of combo boxes for selection. After the selections have been made and the Search button is clicked, an SQL query is custom created with the particular selections made specified in the *where* clause of the query. If “Any” is selected in a field, then that item about the vehicle is not included in the *where* clause. This is so that vehicles that satisfy anything for that field are return, which is what will be required. The code used for setting the clause for the make field is shown below, the result will then be used in the query:

```
if (makeBox.getSelectedItem().equals("Any")) {
    make = "";
} else {
    make = "AND v.make = '" + makeBox.getSelectedItem() + "' ";
}
```

For example if the customer specifies a 4 door saloon car, which has manual transmission, less than 80,000 miles, and is priced between £2,000 to £4,000 (as selected above), the vehicles that satisfy these requirements are displayed in the vehicle stock table (as shown below).

VEHICLE STOCKLIST

Reg Num	Make	Model	Version	Colour	Price (£)	Days in Stock
GF51TVB	Citroen	C5	CD	Grey	4000	43
P468VBT	Ford	Mondeo	LX	Red	2500	61
R638BGT	Honda	Accord	SE	White	2250	33
T389TCK	Toyota	Avensis	FS	Black	2950	62

Search results displayed in table

Search Previous Sale



The search previous sale functionality brings up a small dialogue box (shown above) which was created with a *JOptionPane* component in Java. The code that creates this is:

```
String reg = JOptionPane.showInputDialog(null, "Please enter the vehicle  
Registration Number", "Search Previous Sale",  
JOptionPane.INFORMATION_MESSAGE);
```

When the vehicle registration is entered and OK is clicked, the sale appears in the previous sales table.

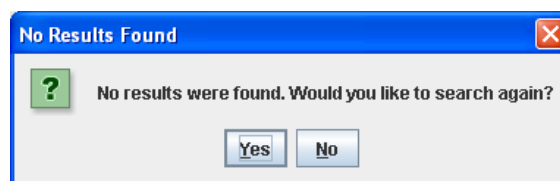
PREVIOUS SALES			
Reg Num	Vehicle	Customer	Date Sold
YK05RST	Volkswagen Golf	Mr. Thomas Phillips	2007-04-11

Usually, users insert a space when entering a registration number, however the database stores registrations without spaces in them. Therefore a *StringTokenizer* had to be used to ensure that if the user does enter a registration with a space, then the spaces from the input are removed before searching the database. The code for this consisted of the following:

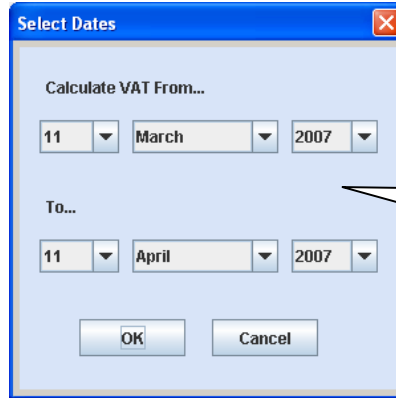
```
StringTokenizer string = new StringTokenizer(reg);  
reg = "";  
while (string.hasMoreTokens()) {  
    reg = reg + string.nextToken();  
}
```

If the vehicle registration number entered in the dialogue box does not match any registration number in the database for previously sold vehicles, then the message shown below is displayed. This prompts the user that no results were found and they are given the option to try a new search. If Yes is clicked, then the search box appears again. The code for this consists of the following, *list* is an *ArrayList* and usually consists of all the sales in the table. If there are 0 sales, then the message appears:

```
if (list.size() == 0) {  
    int choice = JOptionPane.showConfirmDialog(null, "No results were  
found. Would you like to search again?", "No Results Found",  
JOptionPane.YES_NO_OPTION, JOptionPane.QUESTION_MESSAGE);  
    if (choice == JOptionPane.YES_OPTION) {  
        searchSale();  
    }  
}
```



Calculate Total VAT Due

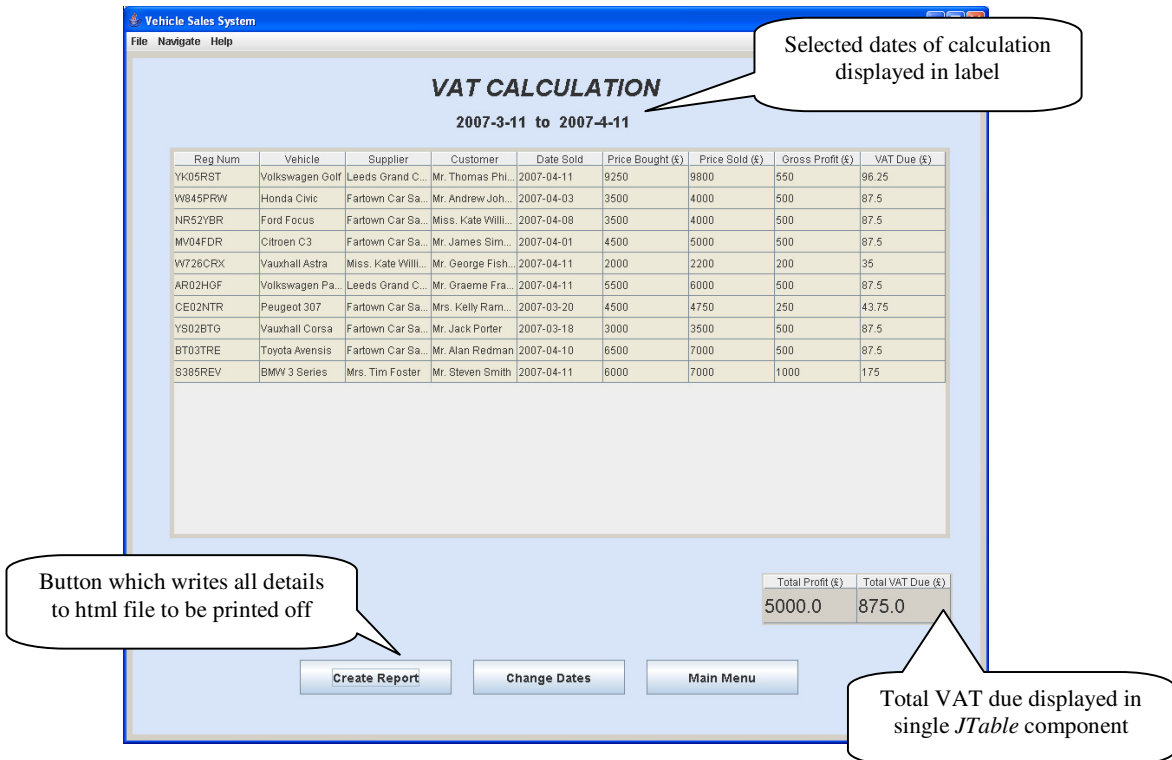


A dialog box titled "Select Dates" with a close button (X) in the top right corner. It contains two sections: "Calculate VAT From..." and "To...". Each section has three dropdown menus for day, month, and year. The "From" section is set to 11, March, 2007. The "To" section is set to 11, April, 2007. At the bottom are "OK" and "Cancel" buttons.

Window created using *JDialog* component, this creates a blank window which can then be customised

When the Calculate VAT button is clicked on the main menu screen, the user is taken to the VAT calculation screen and is prompted with the dates selection box (shown above). This was created using a *JDialog* component which allows a lightweight window component to be custom created. When OK is clicked all the vehicles sold within the dates specified are displayed in the VAT table (as shown below). This screen also includes a total VAT due field in the bottom right of the screen. The code that was assigned to the OK button to perform the action of calculating the VAT on vehicles sold is:

```
vatDateLabel.setText(from + " to " + to);
createVATTable(from, to);
createTotalTable();
vatPanel.remove(scrollPane);
vatPanel.remove(totalScrollPane);
createScrollPane();
createTotalPane();
```



The main window titled "Vehicle Sales System" with a menu bar (File, Navigate, Help). The central area is titled "VAT CALCULATION" and shows the selected dates "2007-3-11 to 2007-4-11". Below this is a table of vehicle sales data. At the bottom right, a summary table shows "Total Profit (£)" as 5000.0 and "Total VAT Due (£)" as 875.0. At the bottom are three buttons: "Create Report", "Change Dates", and "Main Menu".

Reg Num	Vehicle	Supplier	Customer	Date Sold	Price Bought (£)	Price Sold (£)	Gross Profit (£)	VAT Due (£)
YK05RST	Volkswagen Golf	Leeds Grand C...	Mr. Thomas Phil...	2007-04-11	9250	9800	550	96.25
WR45PRW	Honda Civic	Fartown Car Sa...	Mr. Andrew Joh...	2007-04-03	3500	4000	500	87.5
NR52YBR	Ford Focus	Fartown Car Sa...	Miss. Kate Willi...	2007-04-08	3500	4000	500	87.5
MV04FDR	Citroen C3	Fartown Car Sa...	Mr. James Sim...	2007-04-01	4500	5000	500	87.5
W726CRX	Vauxhall Astra	Miss. Kate Willi...	Mr. George Fish...	2007-04-11	2000	2200	200	35
AR02HGF	Volkswagen Pa...	Leeds Grand C...	Mr. Graeme Fra...	2007-04-11	5500	6000	500	87.5
CE02NTR	Peugeot 307	Fartown Car Sa...	Mrs. Kelly Ram...	2007-03-20	4500	4750	250	43.75
YS02BTG	Vauxhall Corsa	Fartown Car Sa...	Mr. Jack Porter	2007-03-18	3000	3500	500	87.5
BT03TRE	Toyota Avensis	Fartown Car Sa...	Mr. Alan Redman	2007-04-10	6500	7000	500	87.5
S385REV	BMW 3 Series	Mrs. Tim Foster	Mr. Steven Smith	2007-04-11	6000	7000	1000	175

Total Profit (£)	Total VAT Due (£)
5000.0	875.0

Printout of VAT Due Calculation for Selected Dates

STEVE GRAVES MOTORS

Unit 1A Huddersfield Road, Birstall, WF17 9AY
TEL: 01924 475382, Fax: 01924 475282

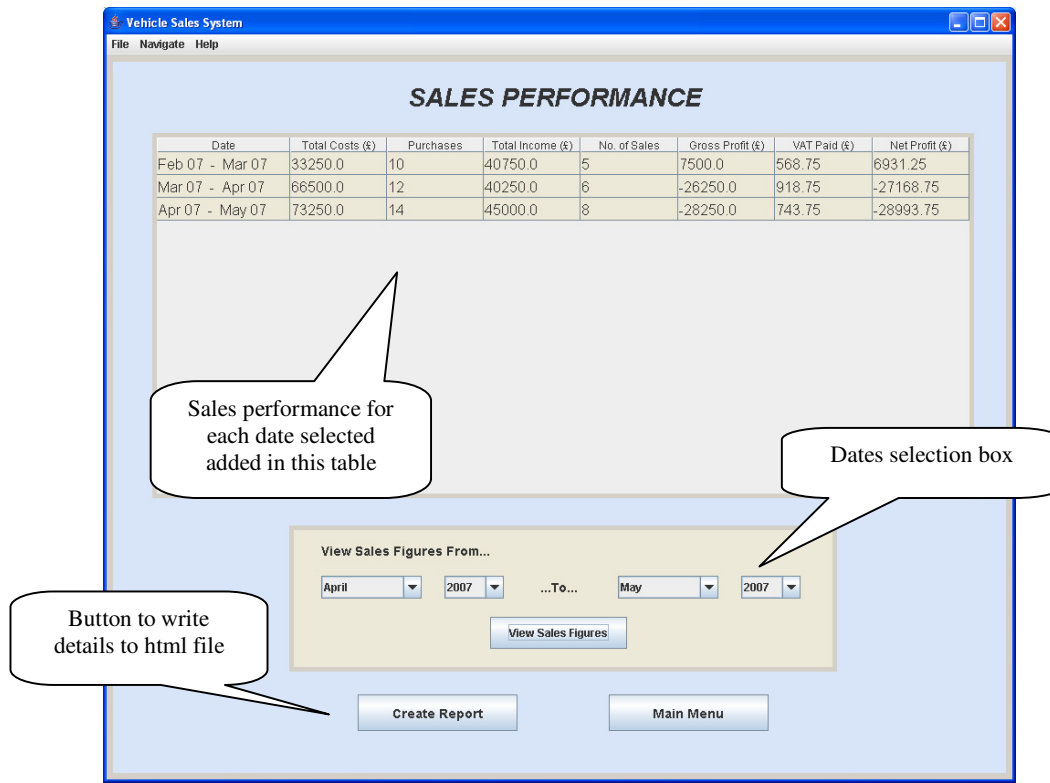
TOTAL VAT DUE REPORT

2007-3-11 to 2007-4-11

REG	VEHICLE	SUPPLIER	CUSTOMER	DATE SOLD	BOUGHT (£)	SOLD (£)	PROFIT (£)	VAT (£)
YK05RST	Volkswagen Golf	Leeds Grand Car Auctions	Mr. Thomas Phillips	2007-04-11	9250	9800	550	96.25
W845PRW	Honda Civic	Fartown Car Sales	Mr. Andrew Johnson	2007-04-03	3500	4000	500	87.5
NR52YBR	Ford Focus	Fartown Car Sales	Miss. Kate Williams	2007-04-08	3500	4000	500	87.5
MV04FDR	Citroen C3	Fartown Car Sales	Mr. James Simmons	2007-04-01	4500	5000	500	87.5
W726CRX	Vauxhall Astra	Miss. Kate Williams	Mr. George Fisher	2007-04-11	2000	2200	200	35
AR02HGF	Volkswagen Passat	Leeds Grand Car Auctions	Mr. Graeme Francis	2007-04-11	5500	6000	500	87.5
CE02NTR	Peugeot 307	Fartown Car Sales	Mrs. Kelly Ramsdon	2007-03-20	4500	4750	250	43.75
YS02BTG	Vauxhall Corsa	Fartown Car Sales	Mr. Jack Porter	2007-03-18	3000	3500	500	87.5
BT03TRE	Toyota Avensis	Fartown Car Sales	Mr. Alan Redman	2007-04-10	6500	7000	500	87.5
S385REV	BMW 3 Series	Mrs. Tim Foster	Mr. Steven Smith	2007-04-11	6000	7000	1000	175

TOTAL PROFIT (£)	TOTAL VAT DUE (£)
5000.0	875.0

View Sales Performance of Business



The combo boxes in the date selection box allows the user to select the months they wish to view the sales figures for, and then by clicking the View Sales Figures button, the figures are displayed in a single row in the table. For each date selected (with the View Sales Figures button being clicked), the figures for that date are added into the table along with the sales figures for the other dates previously selected. The SQL query and code to retrieve and calculate the total costs and income is as follows:

```
ResultSet costResults, salesResults = null;
costResults = db.submitQuery("SELECT p.price_bought FROM pricing AS p,
vehicle AS v WHERE (v.date_bought" + " BETWEEN '" + from + "-01' AND '" +
to + "-01') AND (p.reg = v.reg)");
while (costResults.next()) {
    totalCosts = totalCosts + costResults.getDouble("price_bought");
    totalPurchases = totalPurchases + 1;
}
salesResults = db.submitQuery("SELECT p.price_sold, p.profit, p.vat FROM
pricing AS p, vehicle AS v " + "WHERE (v.date_sold BETWEEN '" + from + "-
01' AND '" + to + "-01') AND (p.reg = v.reg)");
while (salesResults.next()) {
    totalIncome = totalIncome + salesResults.getDouble("price_sold");
    totalSales = totalSales + 1;
    totalVat = totalVat + salesResults.getDouble("vat");
    String vatString = df.format(totalVat);
    totalVat = Double.parseDouble(vatString);
}
```

Printout of Sales Performance for Business

STEVE GRAVES MOTORS

Unit 1A Huddersfield Road, Birstall, WF17 9AY

TEL: 01924 475382, Fax: 01924 475282

SALES REPORT

11-04-2007

DATE	TOTAL COSTS (£)	PURCHASES	TOTAL INCOME (£)	NO. OF SALES	GROSS PROFIT (£)	VAT PAID (£)	NET PROFIT (£)
Feb 07 - Mar 07	33250.0	10	40750.0	5	7500.0	1312.5	6187.5
Mar 07 - Apr 07	36500.0	12	45250.0	6	8750.0	1513.25	7218.75
Apr 07 - May 07	40500.0	14	49750.0	8	9250.0	1618.75	7631.25

Invoice Vehicle Scheduled with Specific Work Carried out and Parts Fitted

Vehicle Sales System
File Navigate Help

EDIT APPOINTMENT FOR VR51DTN

Customer & Vehicle Work Required Work Performed Pricing

Work Carried Out & Parts Fitted:	Price (£):
Regular service carried out	80.00
Fitted new rear brake pads	40.00
Replaced rear brake light	10.00
Applied new bolt and oil to door hinge	15.00

* Required Field

Update Appointment Cancel

Text fields to enter specific work carried out

Text fields to enter price of specific work

The text fields in the Work Performed tab allow data of specific work carried out to be entered, along with the price of each item. The Price fields were set to allow only numerical values including decimal points. Up to seven distinct details about work carried out can be entered. After the individual price fields are filled in, and the user clicks the Calculate Total Price button in the Pricing tab, the total price fields are automatically calculated and filled in, which the user does not have to do manually. The code to calculate the total price and VAT fields in the Pricing tab is shown on the right. Before setting the values to two decimal places for the penny, the value first had to be parsed into a string, and then converting into a decimal.

Customer & Vehicle Work Required Work Performed Pricing

Calculate Total Price

Price Excluding VAT: £ 145.00

VAT at 17.5%: £ 25.38

Total Price: £ 170.38

Fields automatically get updated when button above is clicked

```
exVatPrice = price1 + price2 + price3 + price4 +
price5 + price6 + price7;
String exVatString = df.format(exVatPrice);
exVatPrice = Double.parseDouble(exVatString);
exVatField.setText(exVatString);

vat = ((exVatPrice / 100) * 17.5);
String vatString = df.format(vat);
vat = Double.parseDouble(vatString);
vatField.setText(vatString);

totalPrice = exVatPrice + vat;
String totalPriceString = df.format(totalPrice);
totalPrice = Double.parseDouble(totalPriceString);
totalPriceField.setText(totalPriceString);
```

Printout of Invoice for Work Carried out on a Vehicle that was Scheduled

STEVE GRAVES MOTORS

Unit 1A Huddersfield Road, Birstall, WF17 9AY
TEL: 01924 475382, Fax: 01924 475282

VEHICLE SERVICE INVOICE

2007-04-20

CUSTOMER

Name: William Clarkson
Phone: 07815032729

VEHICLE

Reg No: VR51DTN
Make: Ford
Model: Focus
Engine Size: 1600 cc

WORK CARRIED OUT & PARTS FITTED

PRICE

Regular service carried out	£80
Fitted new rear brake pads	£40
Replaced rear brake light	£10
Applied new bolt and oil to door hinge	£15

Price Excluding VAT:	£145
VAT at 17.5%:	£25.38
Total Price:	£170.38

Login Access

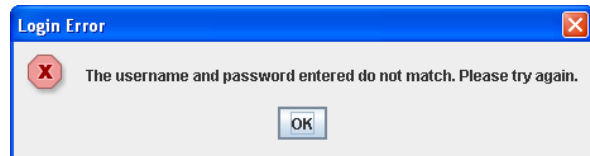


Vehicle Sales System Login

LOGIN

Username:

Password:



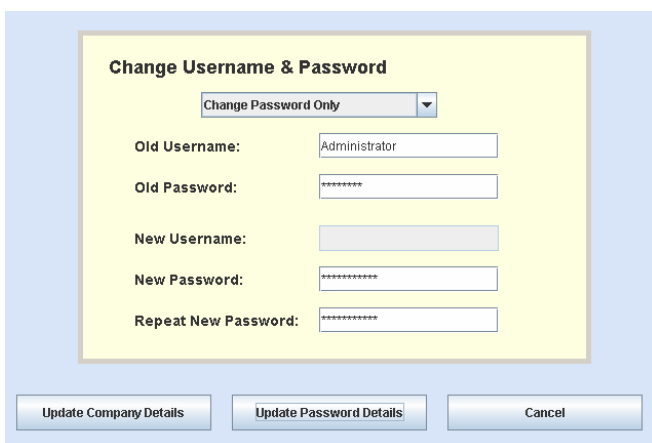
Login Error

 The username and password entered do not match. Please try again.

When the system is run, the login box appears first (above left) prompting the user to enter the system's username and password. When the Login button is clicked, the system checks the entry with the username and password data in the database to see if they match. If they are correct then the main menu of the system is presented. The code to check the credentials was implemented as follows:

```
results = db.submitQuery("SELECT * FROM login");
while (results.next()) {
    username = results.getString("username");
    password = results.getString("password");
}
if(usernameEntry.equals(username) && passwordEntry.equals(password)) {
    pass = "yes";
}
if (pass.equals("yes")) {
    switchPanels(mainMenu.returnMenuPanel());
    loginDialog.setVisible(false);
}
```

If the entry from the user and the username and password data from the database do not match, an error message as shown above right is displayed. When in the system, on the administration screen (below left) the user can change the username and password. If the new password entry does not match with the repeated entry, an error message as shown below right is displayed.



Change Username & Password

Change Password Only

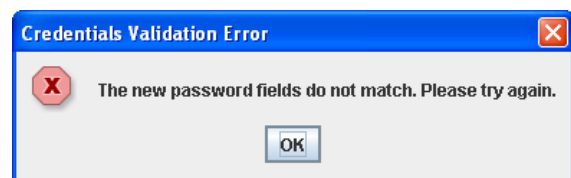
Old Username:

Old Password:

New Username:

New Password:

Repeat New Password:



Credentials Validation Error

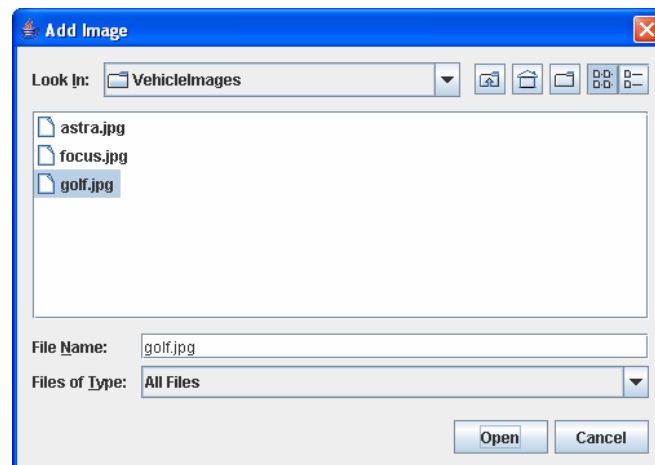
 The new password fields do not match. Please try again.

Add Image of Vehicle

The screenshot shows a web form titled 'Add Image of Vehicle'. It has three tabs: 'Vehicle', 'Image', and 'Supplier'. The 'Image' tab is active. The form contains a 'Vehicle Notes' section with a text area containing the text: 'Full service history, alloy wheels, air conditioning, leather seats, 12 months road tax, 6 months MOT'. A callout points to this text area, stating: 'Text Area to allow larger area for data entry, created using *JTextArea*'. Below the notes is a 'Vehicle Image' section with an 'Add Image' button. A callout points to this button, stating: 'When clicked, File Chooser window will appear'. Below the button is a large image of a silver Volkswagen Golf. A callout points to this image, stating: 'Image of vehicle added'. At the bottom right of the form, there is a small asterisk and the text '* Required Field'.

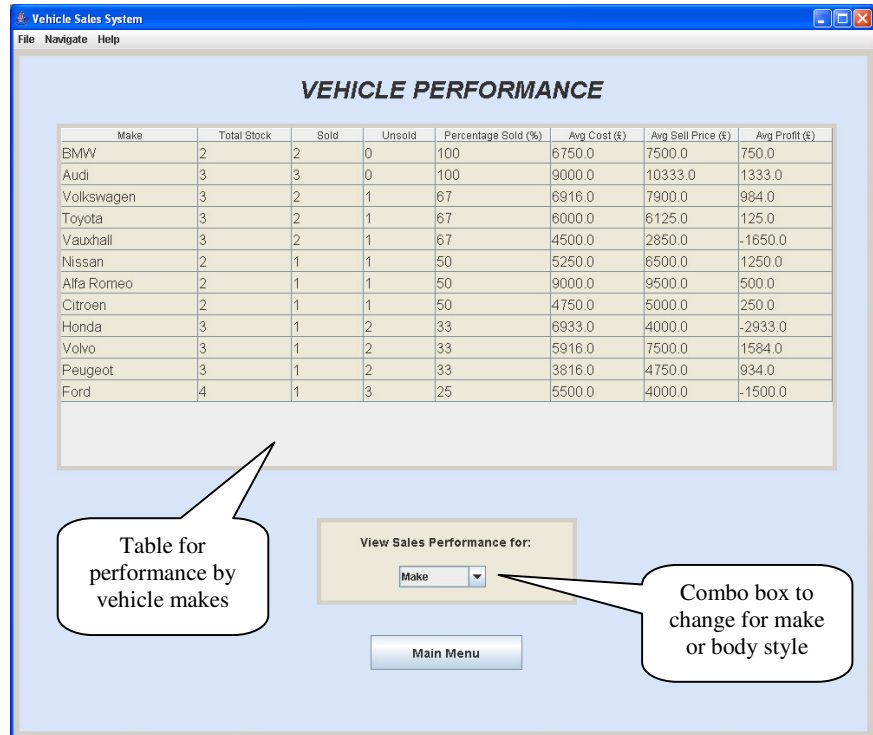
For the functionality to add an image of a vehicle, the user is required to click the Add Image button which then brings up a new window (shown below) where the image file can be selected. This feature was created using a *JFileChooser* instance in Java. The code for this consists of the following when the Add Image button is clicked:

```
JFileChooser fc = new JFileChooser();  
fc.setDialogTitle ("Add Image");  
fc.showOpenDialog(fc);  
File file = fc.getSelectedFile();  
fileName = vehicleImagePath + fc.getName(file);  
ImageIcon image = new ImageIcon(fileName);  
vehicleImage.setIcon(image);
```



View Sales Performance of Vehicles

The screen for viewing the sales performance of vehicles mainly consists of a table that displays the information. A combo box below allows the user to select whether they wish to see the performance on vehicle makes or body styles. As the combo box is changed, the contents of the table changes between makes and body styles without having to click a button. This is done by adding an *ActionListener* class to the combo box component, so that each time an action is performed on that box (changing the selection), then an action is performed, in this case changing the contents of the table.



Vehicle Sales System
File Navigate Help

VEHICLE PERFORMANCE

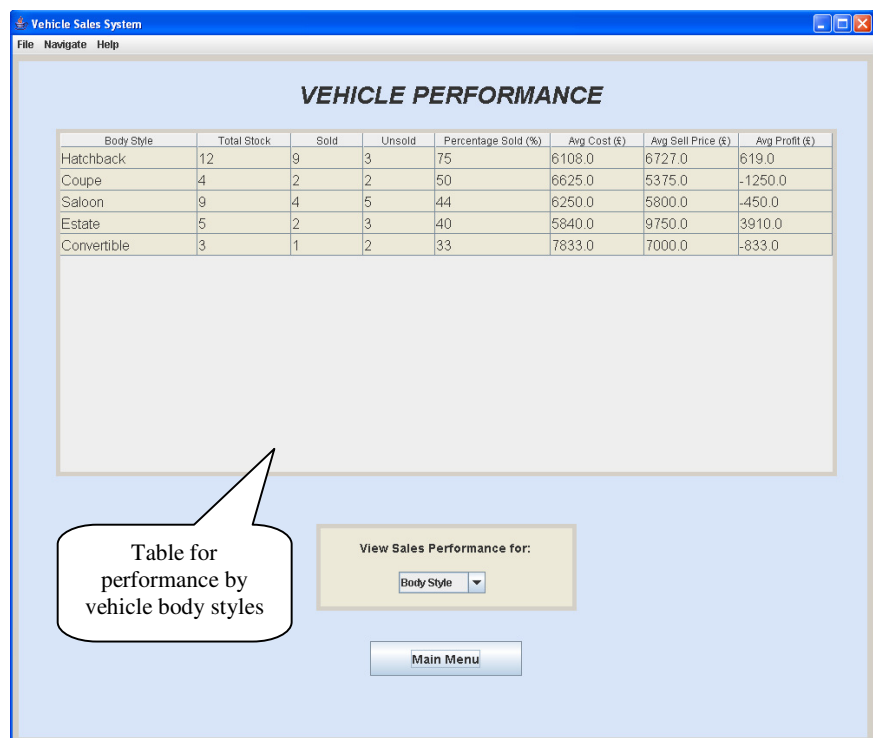
Make	Total Stock	Sold	Unsold	Percentage Sold (%)	Avg Cost (£)	Avg Sell Price (£)	Avg Profit (£)
BMW	2	2	0	100	6750.0	7500.0	750.0
Audi	3	3	0	100	9000.0	10333.0	1333.0
Volkswagen	3	2	1	67	6916.0	7900.0	984.0
Toyota	3	2	1	67	6000.0	6125.0	125.0
Vauxhall	3	2	1	67	4500.0	2850.0	-1650.0
Nissan	2	1	1	50	5250.0	6500.0	1250.0
Alfa Romeo	2	1	1	50	9000.0	9500.0	500.0
Citroen	2	1	1	50	4750.0	5000.0	250.0
Honda	3	1	2	33	6933.0	4000.0	-2933.0
Volvo	3	1	2	33	5916.0	7500.0	1584.0
Peugeot	3	1	2	33	3816.0	4750.0	934.0
Ford	4	1	3	25	5500.0	4000.0	-1500.0

Table for performance by vehicle makes

View Sales Performance for:
Make

Main Menu

Combo box to change for make or body style



Vehicle Sales System
File Navigate Help

VEHICLE PERFORMANCE

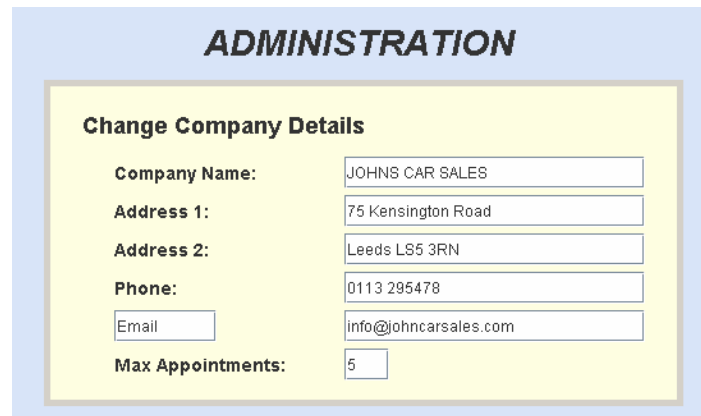
Body Style	Total Stock	Sold	Unsold	Percentage Sold (%)	Avg Cost (£)	Avg Sell Price (£)	Avg Profit (£)
Hatchback	12	9	3	75	6108.0	6727.0	619.0
Coupe	4	2	2	50	6625.0	5375.0	-1250.0
Saloon	9	4	5	44	6250.0	5800.0	-450.0
Estate	5	2	3	40	5840.0	9750.0	3910.0
Convertible	3	1	2	33	7833.0	7000.0	-833.0

Table for performance by vehicle body styles

View Sales Performance for:
Body Style

Main Menu

Change Company Details



ADMINISTRATION

Change Company Details

Company Name:

Address 1:

Address 2:

Phone:

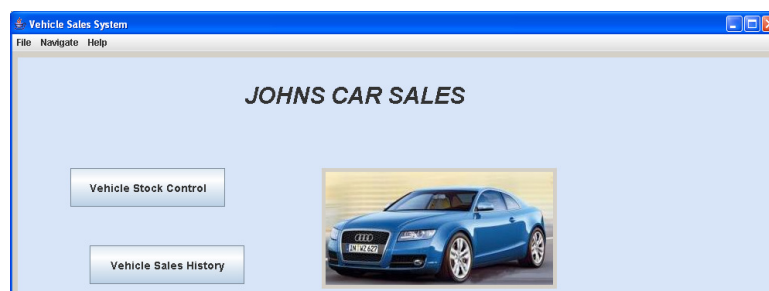
Email:

Max Appointments:

At the administration screen (where the password details can also be changed), the user has the ability to change the company's details. The fields all consist of *JTextField*s with data in them already populated from the database. The user just has to make changes where necessary. When they click the Update Company Details button at the bottom of the screen, the company's details are updated. The code from the method which updates the details in the database consists of the following:

```
db.insertQuery("UPDATE company SET " +  
    "company_name = '" + nameField.getText() +  
    "', address1 = '" + address1Field.getText() +  
    "', address2 = '" + address2Field.getText() +  
    "', phone = '" + phoneField.getText() +  
    "', contact2_type = '" + contact2Field.getText() +  
    "', contact2_value = '" + contact2valueField.getText() +  
    "', max_appointments = '" + appField.getText() +  
    "' ");
```

The changes can be noticed on the main menu screen where the main title of the company name is changed, as well as the details of the company at the top of each printout. These are shown in the diagrams below.



JOHNS CAR SALES
75 Kensington Road, Leeds LS5 3RN
TEL: 0113 295478, Email: info@johncarsales.com

VEHICLE QUOTATION

APPENDIX I – Testing of Second Iteration

Unit Testing for Implementing Search Vehicle

TEST	EXPECTED RESULT	ACTUAL RESULT	COMMENTS
Enter non-alphanumeric characters in 'Registration No' field	Do not allow the non-alphanumeric characters to be entered	As expected	
Select a make from 'Make' box and check whether the models in the 'Model' box are correct for that make	Models in 'Model' box should be the correct models for the make selected only	As expected but models already sold and not for sale were also displayed	Needed to included a <i>where</i> clause in SQL query to specify the sold field must be 'no'
Select a particular make from 'Make' box and click Search	All vehicles of that make should be displayed	As expected	
Select an item from its box apart from price boxes and click Search each time	All vehicles that satisfy the item selected must be displayed in table each time	As expected	
Select a number of price ranges from the price boxes and click Search each time	All vehicles within the price range specified must be displayed in table	Vehicles were not being displayed in table	Minimum price selected was also used as maximum price in SQL query

Unit Testing for Implementing Search Previous Sale

TEST	EXPECTED RESULT	ACTUAL RESULT	COMMENTS
Enter the registration number of a vehicle sold in the input field and click OK	Display the record of the vehicle sold in the table	As expected	
Enter random characters that does not consist of the registration of a vehicle sold in input field and click OK	Display message saying no results were found, and give option of searching again	As expected	
Click 'Yes' when given the option to search another sale when no results found	Display input field again to allow search for a sale	As expected	
Click 'No' when given the option to search another sale when no results found	Go back to table of previous vehicle sales	As expected	

Unit Testing for Implementing VAT Calculation

TEST	EXPECTED RESULT	ACTUAL RESULT	COMMENTS
Click the 'Calculate VAT' button from the main menu	Go to the VAT Calculation screen and display dates selection box	Went to VAT calculation screen but dates selection box was not displayed	The dialog box for selecting dates was not set to be visible when navigated to screen
Select range of dates where vehicles were sold	Display details of the sale and VAT figure for all vehicles sold in that period, with total	As expected	
Click 'Create Report' button when VAT table consists of a number of sales	Create a html file which includes all the details that were displayed on the screen	As expected	
Click 'Change Dates' button and select a range of dates where no vehicles were sold	Display message saying no results were found, and give option of searching again	As expected	
Click 'Yes' when given the option to search another range of dates	Display dates selection box	Nothing happened	Code to call method to display dates selection box was missing when 'Yes' is clicked
Click 'Create Report' button when VAT table is empty	Display message saying report could not be created	As expected	

Unit Testing for Implementing Sales Performance of Business

TEST	EXPECTED RESULT	ACTUAL RESULT	COMMENTS
Click Sales Performance button from the main menu	Display box giving user option to view performance of business or vehicles	As expected	
Click Sales Performance button from option box	Go to screen to view sales performance for business	As expected	
Select a range of months and click View Sales Figures button	Display the sales figures for the months selected	As expected	
Click the View Sales Figures button for the same months selected	Do not display a new record for the same sales figures in the table	A repeat of the sales figures were being displayed	Needed to include an <i>if</i> statement to check that if previously selected dates are selected, then do not display figures
Click 'Create Report' button when tables consist of a number of sales figures	Create a html file which includes all the details that were displayed on the screen	As expected	
Click 'Create Report' button when sales performance table is empty	Display message saying report could not be created	As expected	

Unit Testing for Implementing Invoicing Scheduled Vehicle

TEST	EXPECTED RESULT	ACTUAL RESULT	COMMENTS
Enter any characters apart from numeric or decimal points in the price fields	Do not allow the characters to be entered	As expected	
Click the Calculate Total Price button in Pricing tab after prices entered for work carried out	Fill in the total price and VAT fields in the Pricing tab	As expected	
Populate fields with valid data on edit appointment screen and click Update	Go to view appointment screen with all data that was entered being displayed	As expected	
Click Create Invoice button on the view appointment screen	Create a html file which includes all the details of the vehicle scheduled and work carried out with prices	As expected	

Unit Testing for Implementing Login Access

TEST	EXPECTED RESULT	ACTUAL RESULT	COMMENTS
Enter an incorrect combination of username and password when running the system and click Login	Display error message and do not grant access to the main menu of the system	As expected	
Enter the correct username and password when accessing the system	Grant access to system and display main menu	As expected	
When changing password on Administration screen, enter an incorrect combination for old username and password and click Change Password	Display error message saying the old username and password is incorrect, and do not update password	As expected	
When changing password, enter a value in the Repeat New Password field that is different to value entered in the New Password field	Display error message saying the new passwords entered do not match, and do not update password	Allowed new password to be set	An <i>if</i> statement was not included in code to check passwords match before allowing the password to be changed
When changing password, enter the correct old username and password, and matching values in new password fields	Display message saying new password has been changed, and go back to main menu	As expected	
Run system again and enter new username and password	Grant access to system and display main menu	As expected	

Unit Testing for Implementing Adding Image of Vehicle

TEST	EXPECTED RESULT	ACTUAL RESULT	COMMENTS
Click Add Image button in Image tab on screen for adding new vehicle	Display file chooser window to select an image file	As expected	
Select a image of a vehicle with a .jpg file extension, and click Open	Display image selected in the Image tab	As expected	
Select a file not consisting of a .jpg extension	Do not display anything in Image tab	As expected	
Click Cancel on the file chooser window	Close file chooser window	As expected	

Unit Testing for Implementing View Sales Performance of Vehicles

TEST	EXPECTED RESULT	ACTUAL RESULT	COMMENTS
Click Sales Performance button from the main menu	Display box giving user option to view performance of business or vehicles	As expected	
Click Vehicle Performance button from option box	Go to screen to view sales performance for vehicles	As expected	
Check if all figures in columns were valid	All columns should include valid figures	Percentage column was blank	Percentage field from database was not added to the list of rows
Change option in combo box on screen from 'Make' to 'Body Style'	Change contents of table from sales figures of vehicles by make to body style	Nothing happened	<i>ActionListener</i> method was not added on the combo box to change table when selection is changed

Unit Testing for Implementing Ability to Change Company Details

TEST	EXPECTED RESULT	ACTUAL RESULT	COMMENTS
Change the data in each field in the change company details section on the administration screen and click Update Company Details button	Display message saying that company details have been changed and go to main menu screen	As expected	
Check title on main menu	Title on main menu should be	As expected	

screen, and company info on printouts	the new company name, company info on printouts should be the new details		
---------------------------------------	---	--	--

User Acceptance Testing Feedback

After the implementation of the system in the second iteration, it was presented to the employees of Steve Graves Motors for them to test. The feedback gained from the testing that was carried out by the users is summarised below.

General Comments:

- The images of random parts of cars on the various screens give the system a nice look and make it more professional and presentable.
- The icons used on the tabs allow for clearer, easier recognition, and navigation to the correct tab required.
- The borders around the images and the windows also give a nice look to the interface of the system.
- The feature to search for vehicles with customers requirements works very well, and returns accurate results.
- The information required when calculating the VAT on vehicles sold is all present in the new system. The ease of use of this function is good as it does not require much effort when compared to how it is done in the current system.
- The systems ability to give information on the sales performance of vehicles by make or body style can prove to be very effective in the future. This feature is a lot more useful than initially anticipated.
- Login feature is good to keep the system secure from non staff members.

Improvements and Modifications:

- When calculating the total price of work carried out on a vehicle that was scheduled, include separate prices for the total excluding VAT, the VAT, and the total including VAT, rather than just having a single total price field.
- Include commas in the mileage and price figures in the combo boxes for the search vehicle function to easier differentiate the value of thousands from the other figures.

- When changing the company details, include the ability to change the type of contact rather than having to have a fax number contact all the time.
- On the menu bar, include a navigation to the main menu.

Additional Features:

- On the VAT calculation screen, include a button so that the dates selection box appears again if the dates required to calculate the total VAT needs to be changed.
- Include a feature to be able to print out the sales performance figures for the business when they are viewed on the system.
- The ability to change the username set on the system as well as the password.

APPENDIX J – Evaluation

User Requirements Evaluation Feedback

Choices of opinions for each function:

1. Strongly Agree
2. Agree
3. No Opinion
4. Disagree
5. Strongly Disagree

Function Number	Function	Meets Function Needs	Easy to Perform	No Errors Found	Comments
1	Adding new vehicle	2	2	1	Method of entering date first registered could have been easier
2	Viewing vehicle	2	2	1	Format of date displayed could have been better
3	Editing vehicle	1	2	2	
4	Deleting vehicle	2	1	1	
5	Searching vehicle	3	2	1	Good but does not allow search in additional notes of vehicles
6	Printing vehicle details	2	4	2	Method of writing to file and then printing takes more time and effort
7	Editing supplier	3	1	1	Does not allow ability to delete supplier
8	Processing sale of vehicle	2	3	2	Printing invoice could be easier rather than having to print HTML file
9	Viewing previous sale	3	2	1	Does not show how much vehicle was bought for by company
10	Editing previous sale	4	2	2	Does not provide ability to edit vehicle sold, supplier, and some pricing fields
11	Deleting previous sale	2	1	1	
12	Searching previous sale	2	1	1	Would be good if sale can be searched by customer's name
13	Adding new vehicle for repair/service	2	3	1	Have to check availability of date each time. No ability to see whole schedule of dates with vehicles booked to see which are available
14	Viewing appointment details	2	2	2	

15	Invoicing vehicle with work carried out and parts fitted	3	2	1	Fields for entering information about work carried out or parts fitted were slightly short
16	Cancelling/deleting appointment	1	2	1	
17	Calculating total VAT due	2	2	1	Method of printing out VAT report could have been made better
18	Viewing sales performance of business	2	1	2	Could have had ability to compare dates selected by plotting sales performance figures on graph to allow easier comparison, and more specific statistics
19	Login access	2	1	1	
20	Changing login details	2	2	3	
21	Adding image of vehicle	2	3	1	When window appears to choose image file, it does not automatically locate to directory where images are stored. Need to locate images directory manually
22	Viewing sales performance of vehicles	2	2	4	The average profit for each item should be based on the number sold, not in terms of the total stock whether sold or unsold. Could have included column to show average days in stock
23	Changing company details	2	1	2	
24	User Manual	2	2	2	Could have included section on troubleshooting for encountering problems with system

Usability Evaluation Interview with Steve Graves Motors

1) Was the system easy to learn?

After spending time navigating around the system to see how things can be done, it became clearer how to carry out certain functions. This was mainly due to the clarity of the buttons. The user manual also played a huge part in learning how to use the system quickly as it covered all the functionalities of the system as an easy to follow guide.

2) How efficient was the system to use?

At first, the ability to carry out certain functions was slow due to the unfamiliarity of what to do. But when the same function was carried out a few times, it became a lot more efficient in doing it the next time. The speed of carrying out tasks also improved.

3) Was the usability of the system easy to remember?

The system carries out some functions in similar ways to others, such as viewing and editing a vehicle or sale. So this makes it easier to remember how to perform a function. Once the system was used a few times, it became very familiar in how to carry out a function again.

4) How good was the system at preventing errors?

It was good at preventing accidental errors made by users when performing important functions like deleting a vehicle or sale because it provided confirmation boxes. The feature of checking the availability of a date when scheduling a vehicle was also good so that errors in overbooking a date are prevented. There were also no errors in the data that was entered and later retrieved.

5) Was the system subjectively pleasing?

Yes, compared to carrying out the activities in the current system, the new computerised system is much easier to use, efficient, less time consuming, and offers more capabilities to the business. If a few of the final changes to the system required are developed, there is a high chance that this system could replace the old system in carrying out the daily business activities.

Usability Evaluation Interview with RSC Auto Centre

1) Was the system easy to learn?

Initially it was not very easy to learn due to not knowing what to expect from the system. After a while it got a bit more familiar but still took time in thinking how to perform the functions. It also took time to learn how to navigate around the system, and finding out what the results of a particular action will be.

2) How efficient was the system to use?

When carrying out a function, in most cases it proved to be efficient and quick. As more practice with the system was gained, specific tasks were able to be carried out quicker than before. But the time it took to learn the system meant efficiency was low at first, which later on improved.

3) Was the usability of the system easy to remember?

Some of the simple functions of the system were easy to remember how they are performed like viewing sales performance. The more complex functions required time to think how it is carried out. Some small important aspects of the system were quite easily forgotten at times, maybe the important aspects of the system could have been made clearer.

4) How good was the system at preventing errors?

The system was generally good at preventing errors. A few deliberate mistakes were made on some of the functions of the system see if it would accept it, which it didn't. This is useful in case human errors are accidentally made, and a good feature of the system.

5) Was the system subjectively pleasing?

There was not enough time to use the system to make a proper decision on whether it is subjectively pleasing. It did take time to learn how to use the system and when trying to remember how to perform some functions. The efficiency was good, and if more time was given to use the system, it could have been subjectively pleasing.

APPENDIX K – User Manual

VSS USER MANUAL

2006/2007

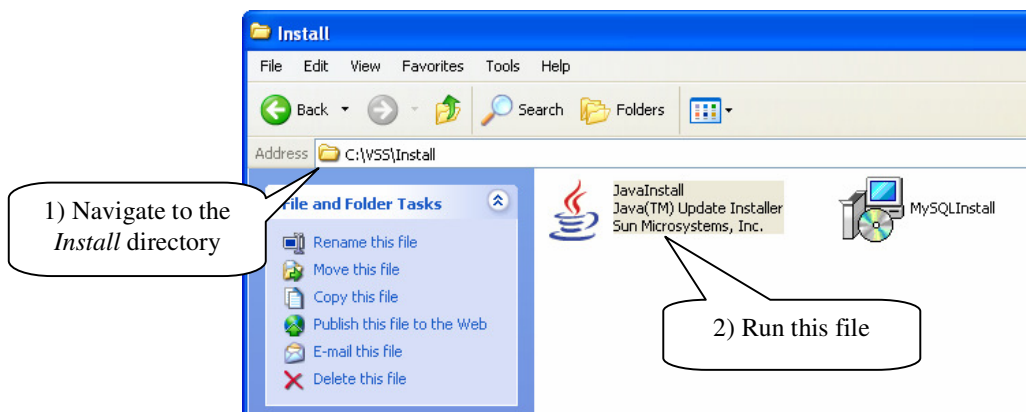
CONTENTS

INSTALLATION.....	1
RUNNING THE SOFTWARE.....	3
SYSTEM LOGIN.....	4
CHANGING USERNAME & PASSWORD.....	4
ADDING NEW VEHICLE.....	5
ADDING IMAGE OF VEHICLE.....	6
EDITING SUPPLIER.....	7
VIEW DETAILS OF VEHICLE.....	7
EDIT DETAILS OF VEHICLE.....	8
PRINT DETAILS OF VEHICLE.....	9
DELETE VEHICLE.....	10
SEARCH VEHICLE.....	10
SELL VEHICLE.....	11
VIEW PREVIOUS SALE DETAILS.....	13
EDIT PREVIOUS SALE DETAILS.....	14
RE-PRINT INVOICE OF PREVIOUS SALE.....	14
DELETE PREVIOUS SALE.....	14
SEARCH PREVIOUS SALE.....	15
ADD NEW APPOINTMENT FOR VEHICLE REPAIR / SERVICE.....	15
VIEW APPOINTMENT DETAILS.....	16
EDIT / INVOICE APPOINTMENT.....	17
CANCEL / DELETE APPOINTMENT	19
CALCULATE VAT DUE.....	19
VIEW SALES PERFORMANCE OF BUSINESS.....	20
VIEW SALES PERFORMANCE OF VEHICLES.....	21
CHANGING COMPANY DETAILS.....	22

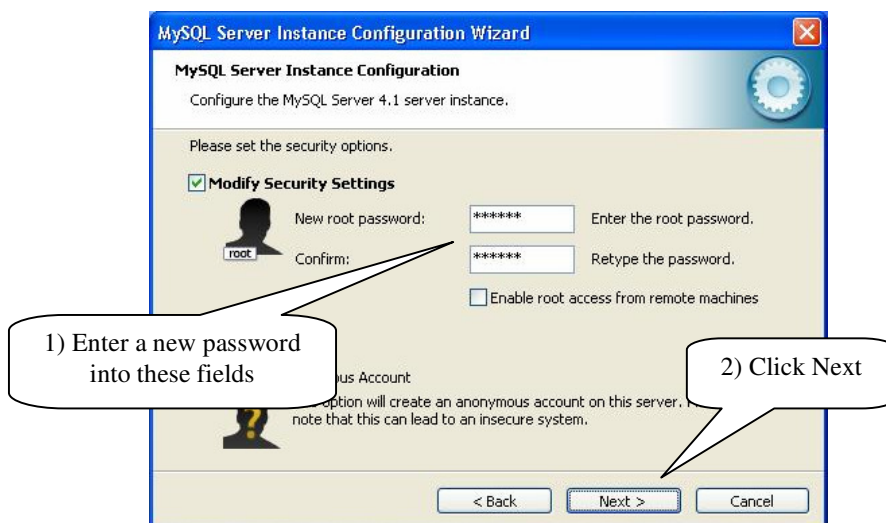
INSTALLATION

Insert the CD into the CD drive, open the CD directory, and copy the folder named *VSS* to a place somewhere on the computer's hard disk.

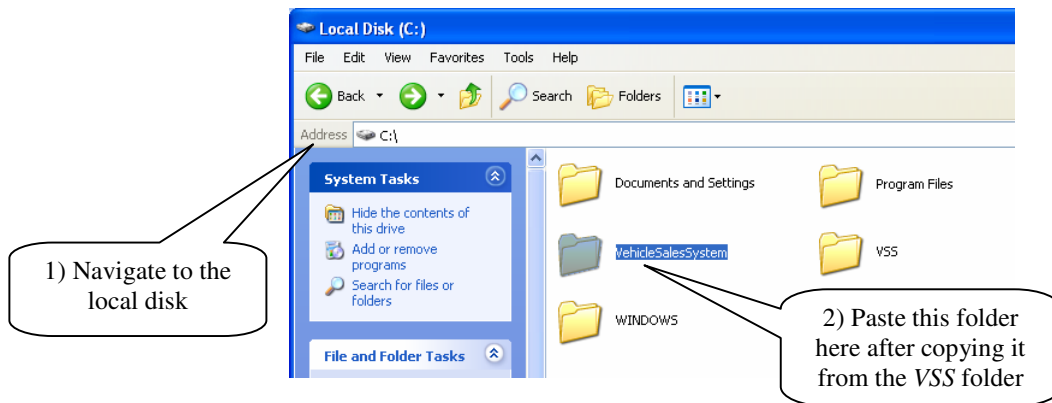
Next, go into the *VSS* folder copied onto the computer's hard disk, and then go into the *Install* directory. Run the *Java Install* file in order to install the Java Runtime Environment required to run the software application.



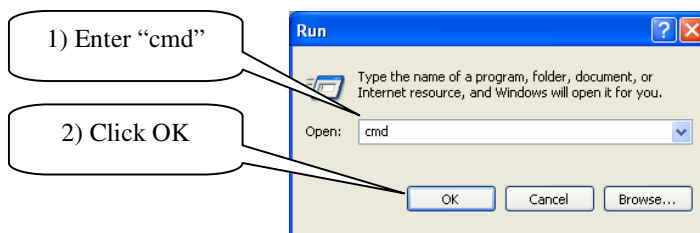
Once Java has been installed, MySQL then has to be installed. Again from the *Install* directory, run the *MySQLInstall* file in order to install MySQL. During the installation, follow the simple wizards to assist in the installation, and make sure that a Typical installation has been made. Skip the option to register with MySQL, and at the final wizard screen, ensure that “Configure” is selected in order to configure the MySQL settings. When the configuration wizard is run, ensure that Standard configuration is selected. At the screen that asks for entering a root password, enter a valid password and ensure that this is remembered and kept safe. At the final screen, click “Execute” to finish the configuration. MySQL will now be installed.



Now go back to the *VSS* directory, copy the *VehicleSalesSystem* folder and paste it into the local disk of the computer. So if you are using the software on a Microsoft Windows operating system, copy the folder *VehicleSalesSystem* directly into the C: drive.



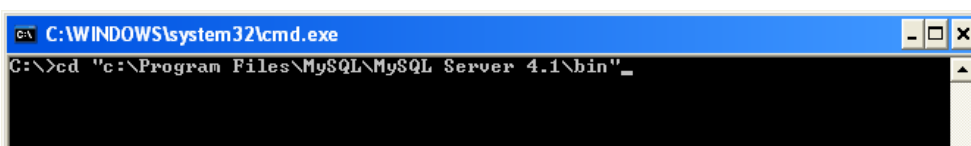
The database tables required to store the data entered into the software application need to be created in MySQL. To do this, first open a command prompt by going to *Start-Run*, enter “cmd” in the run box, and click OK.



When the command prompt appears, enter the command to navigate to the *bin* directory where MySQL was stored. If this was done on a Windows machine, it will probably consist of:

```
cd c:\Program Files\MySQL\MySQL Server 4.1\bin
```

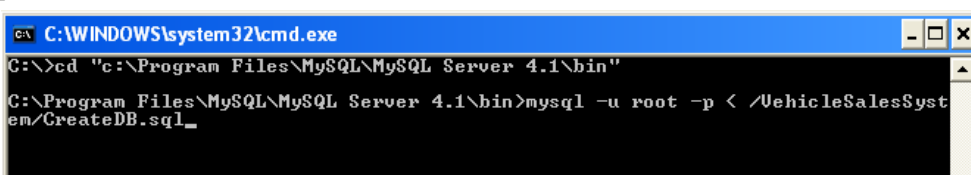
Then press Enter.



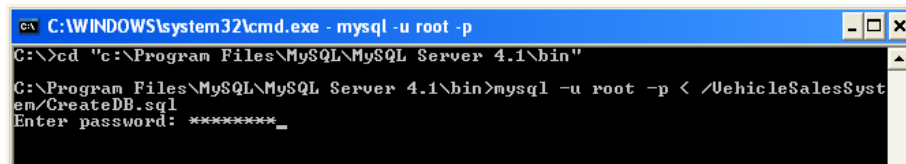
Next enter the following:

```
mysql -u root -p < /VehicleSalesSystem/CreateDB.sql
```

Then press Enter.

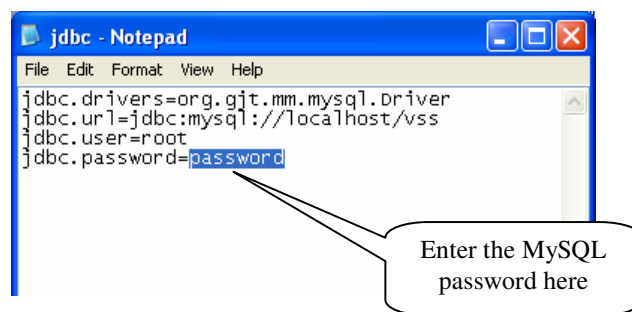


When prompted for the password, enter the root password that was set during the configuration of installing MySQL, as previously described. The required tables will now be created.



```
C:\WINDOWS\system32\cmd.exe - mysql -u root -p
C:\>cd "c:\Program Files\MySQL\MySQL Server 4.1\bin"
C:\Program Files\MySQL\MySQL Server 4.1\bin>mysql -u root -p < /VehicleSalesSystem/CreatedDB.sql
Enter password: *****_
```

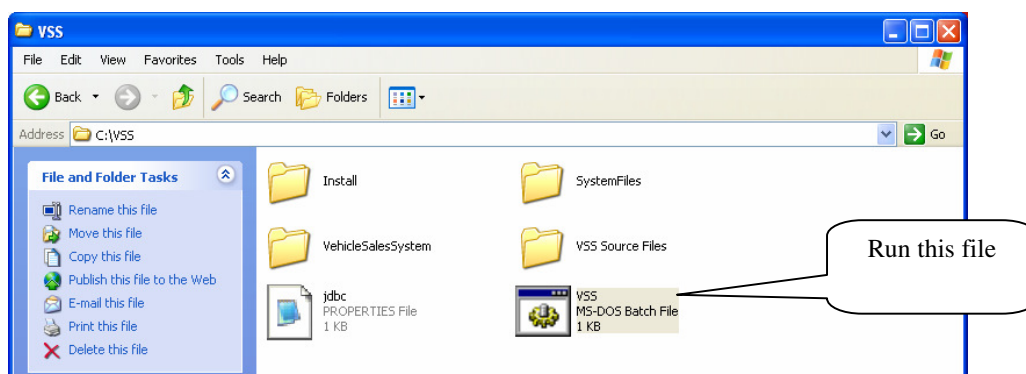
The final part of the installation involves configuring the *jdbc* file in the *VSS* directory. Open the *jdbc* file with a normal text editor such as Notepad (if using Microsoft Windows), and then enter the root MySQL password that was set earlier during the configuration of installing MySQL into the *jdbc.password* line, as shown below. Finally save this file, and close it.



The system is now fully installed and ready to use.

RUNNING THE SOFTWARE

In order to run the software, double-click the *VSS* file found within the *VSS* directory.



SYSTEM LOGIN

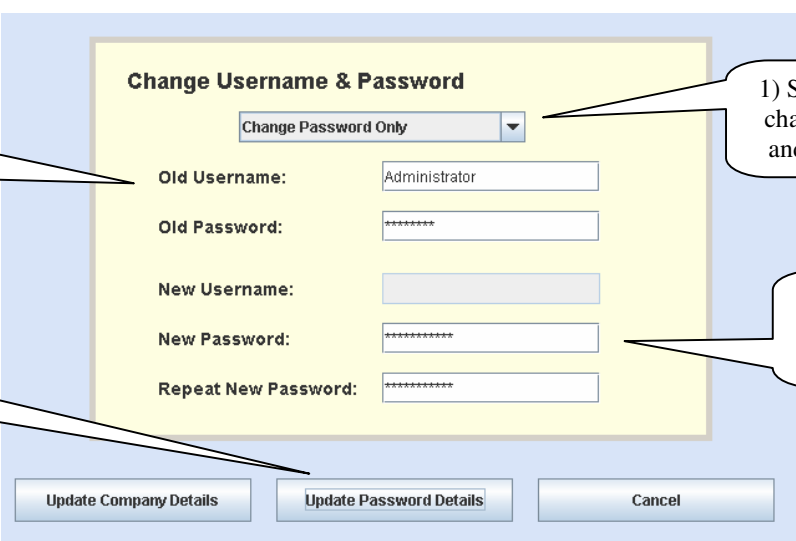
When the system is run, the Login screen will appear. Enter the Username and Password set on the system and click *Login* (as shown below). If the system is run for the first time, the default username is set to “Administrator”, and the password is set to “password”. It is important and highly recommended that the password is changed once the system is accessed for the first time. Details on how to change the username and password is explained in the next section.



The screenshot shows the 'Vehicle Sales System Login' window. It has a title bar with a close button. The main area is titled 'LOGIN'. It contains two text input fields: 'Username:' with the text 'Administrator' and 'Password:' with masked characters '*****'. Below these fields are two buttons: 'Login' and 'Exit'. Two callout boxes provide instructions: one points to the input fields with the text '1) Enter username and password', and the other points to the 'Login' button with the text '2) Click Login button'.

CHANGING USERNAME & PASSWORD

To change the username and password of the system, click the *Administration* button on the main menu to go to the Administration screen. In the Change Username and Password section, select whether both the username and password need to be changed, or just the password from the drop down box. Next enter the current username and password to authenticate the user, and then set the new username and/or password of the system. The new password must be entered twice to validate that it was entered correctly. Finally click the *Update Password Details* button.



The screenshot shows the 'Change Username & Password' window. It has a title bar. The main area is titled 'Change Username & Password'. It contains a dropdown menu with the text 'Change Password Only'. Below this are six text input fields: 'Old Username:' with the text 'Administrator', 'Old Password:' with masked characters '*****', 'New Username:', 'New Password:', 'Repeat New Password:', and another masked password field. At the bottom are three buttons: 'Update Company Details', 'Update Password Details', and 'Cancel'. Four callout boxes provide instructions: one points to the dropdown menu with the text '1) Select option to change username and/or password', one points to the 'Old Username' and 'Old Password' fields with the text '2) Enter current username and password', one points to the 'New Username' and 'New Password' fields with the text '3) Enter new username and /or password', and one points to the 'Update Password Details' button with the text '4) Click Update Password button'.

ADDING NEW VEHICLE

To add a new vehicle to the stock, first click the *Vehicle Stock Control* button from the main menu, and then at the stock list screen click the *Add New Vehicle* button. This will bring up the screen where the details of a vehicle can be entered into the relevant fields provided. The make and model can be selected from the drop down boxes if the same type of vehicle has been previously added. If the make or model does not appear in the boxes, it can be entered into the text field beside it.

Vehicle Image Supplier

*Registration Number: YK05RST

Make: Volkswagen

Model: Golf

Version: GTI

Engine Size: 2000 cc

Colour: Silver

Number of Doors: 5 Doors

Body Style: Hatchback

Fuel Type: Petrol

Transmission: Manual

Mileage: 25000

*Date First Registered: 2005-06-22 (yyyy-mm-dd)

Chassis Number: JD83HD74HFI48DJH47

* Required Field

Enter data into relevant fields

In the Supplier tab, if the vehicle has been supplied by a previously saved supplier, the supplier's name can be selected from the drop down box. The relevant fields will then be automatically filled in. If the supplier is a new supplier, their details can be entered into the relevant fields and if this supplier is required to be saved for potential future supplies, then the checkbox in the tab can be ticked. This will then ensure that the supplier's name appears in the supplier box next time a vehicle is added.

Vehicle Image Supplier

Previous Supplier: Grand Car Auctions

*Name: Grand Car Auctions

Address: 23 Oxford Street

Pudsey

Leeds

Postcode: LS3 7NH

Phone: 0113 645874

Email: info@grandauctions.com

Date: 10 April 2007

*Purchase Price: £ 9250

Selling Price: £ 10000

Save as future supplier

* Required Field

If supplier is previous supplier, their name can be selected from this box

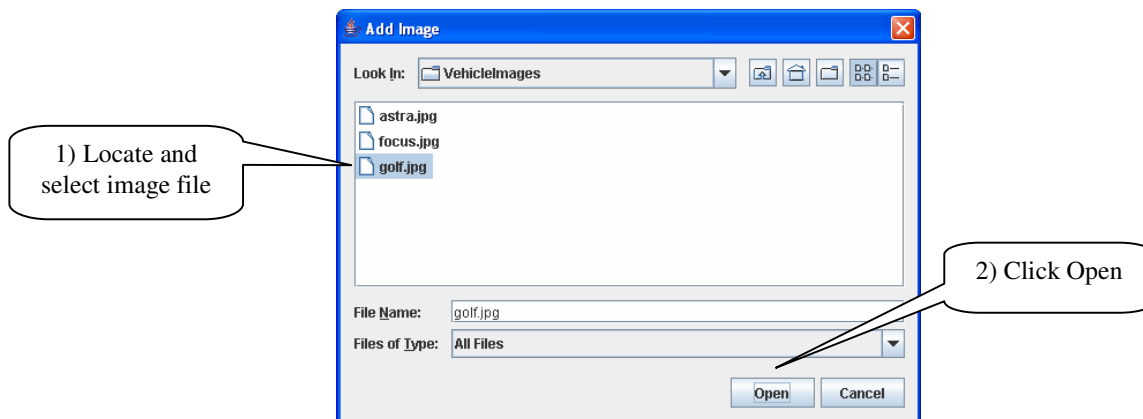
If a new supplier needs to be saved as a future supplier, tick this box

When the required fields are filled in with the relevant data about the vehicle, click the *Add Vehicle* button.

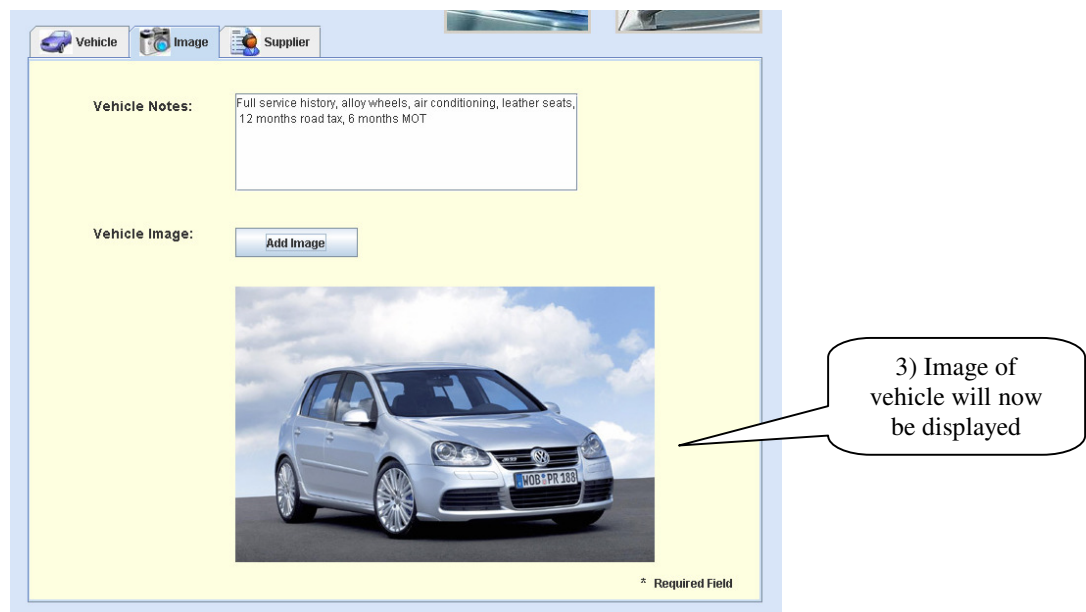


ADDING IMAGE OF VEHICLE

To add an image of a vehicle, the image file must first be placed in the directory *C:\VehicleSalesSystem\VehicleImages*. Then in the Image tab at the Add Vehicle screen, click the *Add Image* button. Locate and select the image file of the vehicle required, and click *Open* in the file chooser window.



The image of the vehicle will now appear in the Image tab on the Add Vehicle screen, as shown below.



EDITING SUPPLIER

To edit a previously saved supplier, click the *Edit Supplier* button on the Add Vehicle screen. This will bring up a new window to edit the supplier's details. From the Original Supplier drop down box, select the supplier's name whose details need to be changed, and their current details will be filled in the fields below. Change the details in the fields as required, and then click the *Update* button. The details of the supplier will not be updated.

The 'Edit Supplier' window contains the following fields and buttons:

- Original Supplier:** A dropdown menu currently showing 'Grand Car Auctions'. Callout: 1) Select previous supplier from drop down box
- New Name:** Text field containing 'Leeds Grand Car Auctions'
- Address:** Three stacked text fields containing '23 Regent Street', 'Headingley', and 'Leeds'. Callout: 2) Change supplier's details as required in the fields
- Postcode:** Text field containing 'LS3 7NW'
- Phone:** Text field containing '0113 645874'
- Email:** Text field containing 'info@grandcarauctions.com'
- Buttons:** 'Update' and 'Cancel' at the bottom. Callout: 3) Click *Update* button

VIEW DETAILS OF VEHICLE

To view a vehicle, click the *Vehicle Stock Control* button from the main menu to go to the Vehicle Stock List screen. Next select the vehicle whose details need to be viewed from the table, and click the *View Vehicle Details* button.

The Vehicle Stock List screen displays a table of vehicles and a set of buttons at the bottom.

T589TCR	Toyota	Avenis	FS	Black	2950	66
S372TBV	Vauxhall	Astra	CC	Red	3250	60
V654FPB	Volkswagen	Passat	E	Green	3250	42
YK05RST	Volkswagen	Golf	GTI	Silver	10000	5
T528TCP	Volvo	C70	SC	Silver	4000	51
R385TPM	Volvo	XC70	S	Grey	3500	47

Callouts:

- 1) Select vehicle from table (pointing to the highlighted row YK05RST)
- 2) Click this button (pointing to the 'View Vehicle Details' button)

Buttons at the bottom: Add New Vehicle, Search Vehicle, View Vehicle Details, Main Menu

The vehicle's details will now be displayed on the screen in the tabs, as shown below.

The screenshot shows a window titled "Vehicle Sales System" with a menu bar (File, Navigate, Help). The main title is "VEHICLE DETAILS FOR YK05RST". Below the title are three tabs: "Vehicle", "Image", and "Supplier". The "Vehicle" tab is active, displaying a list of vehicle details in a yellow box:

- Registration Number: YK05RST
- Make: Volkswagen
- Model: Golf
- Version: GTI
- Engine Size: 2000 cc
- Colour: Silver
- Number of Doors: 5 Doors
- Body Style: Hatchback
- Fuel Type: Petrol
- Transmission: Manual
- Mileage: 25000
- Date First Registered: 2005-06-22
- Chassis Number: JD83HD74HF148DJH47

At the bottom of the window are five buttons: "Edit Vehicle", "Print Details", "Sell Vehicle", "Delete Vehicle", and "Back".

EDIT DETAILS OF VEHICLE

To edit the details of a vehicle, click the *Edit Vehicle* button on the screen for viewing a vehicle. This will then bring up the screen to edit the details of the vehicle. The fields will already be populated with the vehicle's details. Change the details in the field as required, and then click the *Update Vehicle* button. The details of the vehicle will now be updated.

The screenshot shows a window titled "Vehicle Sales System" with a menu bar (File, Navigate, Help). The main title is "EDIT VEHICLE FOR YK05RST". Below the title are three tabs: "Vehicle", "Image", and "Supplier". The "Vehicle" tab is active, displaying a form for editing vehicle details. The form fields are populated with the same data as the previous screen:

- *Registration Number: YK05RST
- Make: Volkswagen
- Model: Golf
- Version: GTI
- Engine Size: 2000 cc
- Colour: Silver
- Number of Doors: 5 Doors
- Body Style: Hatchback
- Fuel Type: Petrol
- Transmission: Manual
- Mileage: 25000
- *Date First Registered: 2005-06-22 (yyyy-mm-dd)
- Chassis Number: JD83HD74HF148DJH47

At the bottom of the window are two buttons: "Update Vehicle" and "Cancel".

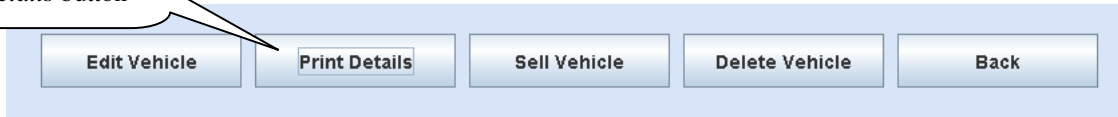
Two callouts are present:

- Callout 1: "1) Change vehicle details in fields" points to the "Number of Doors" field.
- Callout 2: "2) Click Update Vehicle button" points to the "Update Vehicle" button.

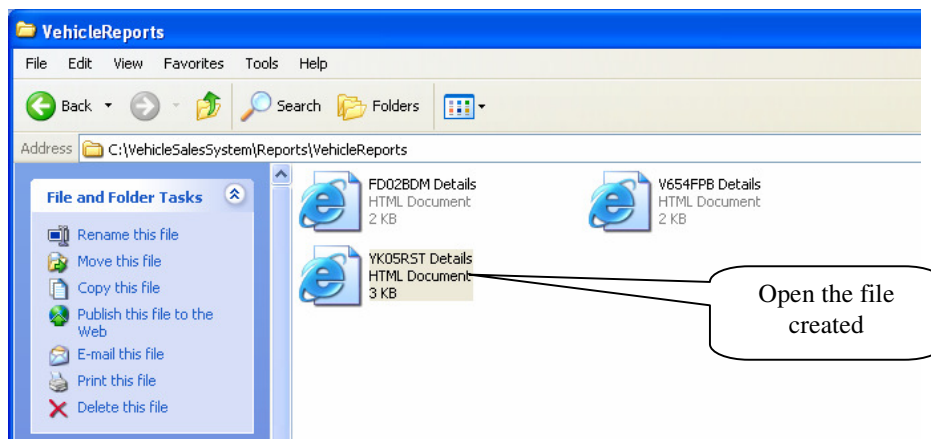
PRINT DETAILS OF VEHICLE

To print the details of a vehicle, click the *Print Details* button from the screen to view a vehicle.

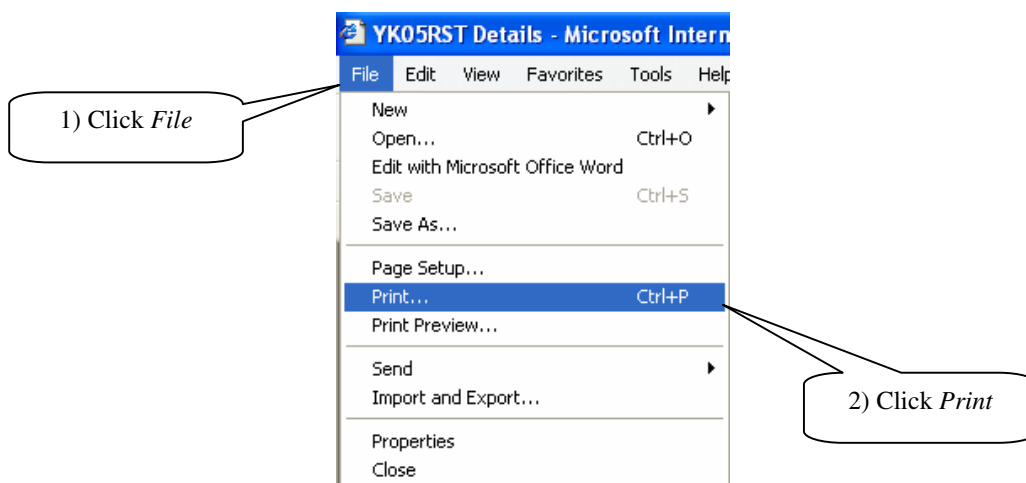
Click *Print Details* button



This will now create a HTML file containing all the details which can be printed off. To print the file, navigate to the following directory *C:\VehicleSalesSystem\Reports\VehicleReport* where the file is created, locate the filename which will include the vehicle's registration number, and open the file.

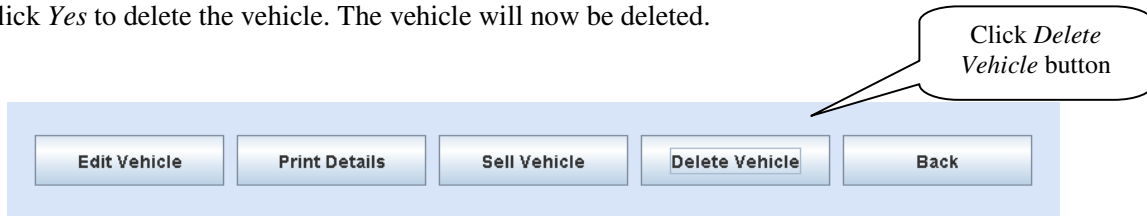


When the file has been opened, click *File* at the top of the screen in the menu bar, and select *Print*. When the Print window appears, click the *Print* button to print the vehicle's details.



DELETE VEHICLE

To delete a vehicle from the stock, go to the screen where the vehicle's details can be viewed, and then click the *Delete Vehicle* button. A message will be displayed to confirm the deletion of the vehicle, click *Yes* to delete the vehicle. The vehicle will now be deleted.



SEARCH VEHICLE

To search for a vehicle from the current stock, click the *Search Vehicle* button on the Vehicle Stock List screen to display the window to select the vehicle requirements. Make the required selections from the fields, and click *Search*. The results of the sale will be displayed in the stock list table.

SEARCH VEHICLE

Registration No:

Make:

Model:

Number of Doors:

Body Style:

Colour:

Fuel:

Transmission:

Maximum Age:

Maximum Mileage:

Price Range: Min: £ Max: £

1) Select vehicle requirements from fields

2) Click *Search* button

SELL VEHICLE

To process the sale of a vehicle, go to the screen to view a vehicle's details and click the *Sell Vehicle* button. On the Vehicle Sale Invoice screen, enter the relevant data in the fields in the Customer, Part-Exchange, and Pricing tabs. In the Customer tab, the Surname field is a required field and therefore data must be entered into this field to process the sale.

The screenshot shows the 'VEHICLE SALE INVOICE' window with the 'Customer' tab selected. The form contains the following fields and options:

- Title: Mr. (dropdown)
- First Name: Thomas
- *Surname: Phillips
- Save as future supplier if part-exchanged: ☐
- Address: 36 Hyde Road, Shipley, Bradford
- Postcode: BD7 4FR
- Phone: 01274 248751
- Email: thomasphillips@email.com

Two callout boxes provide instructions:

- 1) Tick if required to be saved as future supplier (pointing to the checkbox)
- 2) Enter customer details in fields (pointing to the name and address fields)

If the customer wishes to part exchange their old vehicle, then fill in the vehicle's details in the Part-Exchange tab. If the user does not wish to part-exchange their old vehicle, then tick the checkbox at the top of the tab.

☒ Customer does not wish to part-exchange vehicle

In the Pricing tab, all price fields must be entered to process the sale of the vehicle apart from the Part-Exchange Value field if the customer is not part-exchanging their old vehicle. If the customer is paying for the vehicle they are buying by monthly finance, then 'Credit' must be selected from the Method of Payment drop down box. The four finance fields below this must then be filled in. Finally click the *Process Sale* button. A message will be displayed to confirm the process of the sale, click *Yes* to continue with the completion of the sale.

Vehicle Sales System
File Navigate Help

VEHICLE SALE INVOICE

Vehicle Customer Part-Exchange Pricing

Date of Sale: 16 April 2007

Method of Payment: Cash 1) Select the method of payment

Finance Company:

Amount of Loan: £

Duration: months 2) If 'Credit' selected as payment method, fill in these fields

Payment per Month: £

List Price: £ 10000

*Final Price: £ 9800

Part Exchange Value: £ 2000

*Amount to Pay: £ 7800 3) Fill in price fields

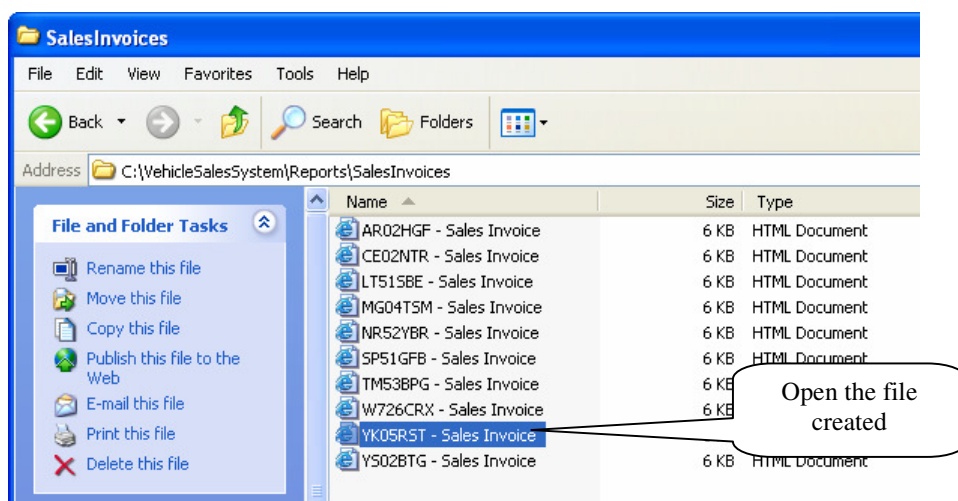
*Amount Paid: £ 100

*Amount Due: £ 7700

* Required Field

Process Sale Cancel 4) Click Process Sale button

A HTML file containing all the details of the sale will now be created. To print the file, navigate to the following directory *C:\VehicleSalesSystem\Reports\SalesInvoices* where the file is created, locate the filename which will include the vehicle's registration number, and open the file. Finally print the file in the same way as printing the details of a vehicle.



VIEW PREVIOUS SALE DETAILS

To view the details of a sale that has been carried out, click the *Vehicle Sales History* button from the main menu to go to the Previous Sales screen. Next select the sale whose details need to be viewed from the table, and click the *View Sale Details* button.

Reg Num	Vehicle	Customer	Date Sold
AR02HGF	Volkswagen Passat	Mr. Graeme Francis	2007-04-11
W726CRX	Vauxhall Astra	Mr. George Fisher	2007-04-11
S385REV	BMW 3 Series	Mr. Steven Smith	2007-04-11
YK05RST	Volkswagen Golf	Mr. Thomas Philips	2007-04-11
BT03TRE	Toyota Avensis	Mr. Alan Redman	2007-04-10
NR52YBR	Ford Focus	Miss. Kate Williams	2007-04-08
WR45BRW	Honda Civic	Mr. Andrew Johnson	2007-04-03

1) Select sale from table

2) Click this button

The sale details will now be displayed on the screen in the tabs, as shown below.

Vehicle Sold	Image	Supplier	Customer	Part-Exchanged	Pricing
Customer Details:					
Name:		Mr. Thomas Philips			
Address:		36 Hyde Road Shipley Bradford			
Postcode:		BD7 4FR			
Phone:		01274 248751			
Email:		thomasphilips@email.com			
Sale Details:					
Date of Sale:		2007-04-11			
Method of Payment:		Card			
Finance Company:					
Amount of Loan:		£ 0			
Duration:		0 months			
Payment per Month:		£ 0			
List Price:		£ 10000			
Price Sold:		£ 9800			
Part Exchange Value:		£ 2000			
Amount to Pay:		£ 7800			
Amount Paid:		£ 100			
Amount Due:		£ 7700			

EDIT PREVIOUS SALE DETAILS

To edit the details of a sale that was carried out, click the *Edit Sale* button on the screen to view a sale's details. This will bring up the screen to edit the sale details. Make the required changes in the tabs, and click the *Update Sale* button. The details of the sale will now be updated.

Vehicle Sales System
File Navigate Help

EDIT SALE DETAILS FOR YK05RST

Vehicle Sold Supplier Customer Part-Exchanged Pricing

Date of Sale: 2007-04-11

Date of Sale: Card

Finance Company:

Amount of Loan: £ 0

Duration: 0 months

Payment per Month: £ 0

List Price: £ 10000

*Final Price: £ 9800

Part Exchange Value: £ 2000

*Amount to Pay: £ 7800

*Amount Paid: £ 100

*Amount Due: £ 7700

* Required Field

1) Make required changes in fields

2) Click *Update Sale* button

Update Sale Cancel

RE-PRINT INVOICE OF PREVIOUS SALE

To print a new copy of a sale that has been carried out, click the *Create Invoice* button on the screen to view the sale's details. A HTML file containing all the details of the sale will now be created. To print the file, navigate to the following directory *C:\VehicleSalesSystem\Reports\SalesInvoices* where the file is created, locate the filename which will include the vehicle's registration number, and open the file. Finally print the file in the same way as printing the initial sale invoice that was created.

DELETE PREVIOUS SALE

To delete the details of a previous sale that was carried out, click the *Delete Sale* button on the screen to view the sale's details. A deletion confirmation box will be displayed, click *Yes* to complete the deletion. The sale will now be deleted.

Edit Sale Create Invoice Delete Sale Back

2) Click *Delete Sale* button

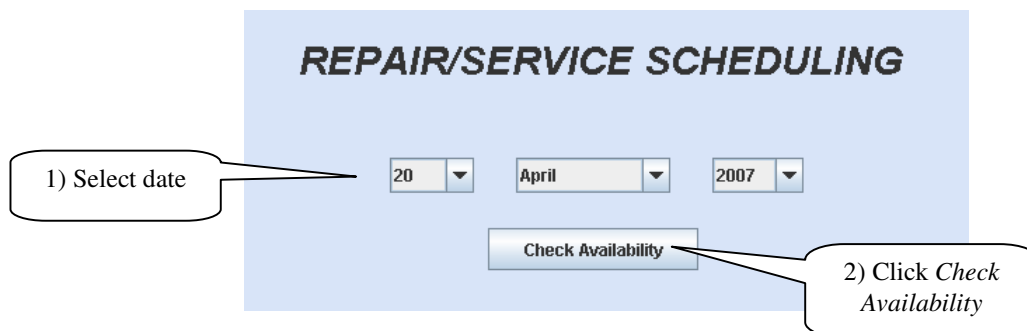
SEARCH PREVIOUS SALE

To search for a previous sale, click the *Search Sale* button on the previous sales screen, which will display a box to enter the registration number of a vehicle that was sold. When the box is displayed, enter the registration number of the vehicle that was sold and click OK. If the sale is found, it will be displayed in the previous sales table. If the sale could not be found, then a message will be displayed asking whether another is required.



ADD NEW APPOINTMENT FOR VEHICLE REPAIR / SERVICE

To schedule a vehicle for an appointment, click the *Vehicle Scheduling* button on the main menu to go to the scheduling screen. Select the date required for the appointment at the top of the screen from the drop down boxes, and click *Check Availability*.



The appointments scheduled on that date will now be displayed in the table. Make sure that the maximum number of appointments allowed on a single day is not already reached for that date, and then click the *Add New Appointment* button.

A new screen will now appear to allow the details of the appointment to be entered into a number of fields. Enter the relevant information in the fields in the Customer & Vehicle tab, and the Work Required tab and then click the *Add Appointment* button. The appointment of the vehicle will now be added and can be viewed in the scheduling table.

NEW APPOINTMENT FOR 20 April 2007

Customer & Vehicle

*Customer Name: William Clarkson

Customer Phone: 07815032729

*Vehicle Reg No: VR51DTN

Vehicle Make: Ford

Vehicle Model: Focus

Engine Size: 1600 cc

Work Required

Type of Work: Service

Date Scheduled: 20 April 2007

Work Requirements: Check brakes and tyres, top up all oils, check exhaust, replace rear brake light, fix door hinge

Add Appointment

1) Enter required information in fields

2) Click Add Appointment button

VIEW APPOINTMENT DETAILS

To view the details of a vehicle that has been scheduled for a repair or service, go to the Vehicle Scheduling screen, select the dates when the appointment was scheduled and click the *Check Availability* button. When the appointment is displayed in the table, select it and click the *View Appointment* button.

Appointments for 20 April 2007

Reg No	Vehicle	Customer	Work Type	Date
VR51DTN	Ford Focus	William Clarkson	Service	2007-04-20

New Appointment View Appointment Main Menu

1) Select appointment from table

2) Click View Appointment button

The appointment details will now appear on the screen in the tabs, as shown below.

APPOINTMENT DETAILS FOR VR51DTN

Vehicle & Customer | Work Required | Work Performed | Pricing

Customer Name: William Clarkson

Customer Phone: 07815032729

Vehicle Reg No: VR51DTN

Vehicle Make: Ford

Vehicle Model: Focus

Engine Size: 1600 cc

The Work Performed and Pricing tabs will be automatically added, but they will initially not have any data in them. Data will only be displayed in them once the appointment has been edited and details about work carried out on the vehicle are added.

EDIT / INVOICE APPOINTMENT

To edit or invoice an appointment with specific work carried out and prices, click the *Edit Appointment* button when viewing the appointments details at the view appointment screen. On the Edit Appointment screen enter into the fields the required information about work that was carried out or parts fitted in the Work Performed tab, along with their prices in the fields next to them.

Work Carried Out & Parts Fitted:

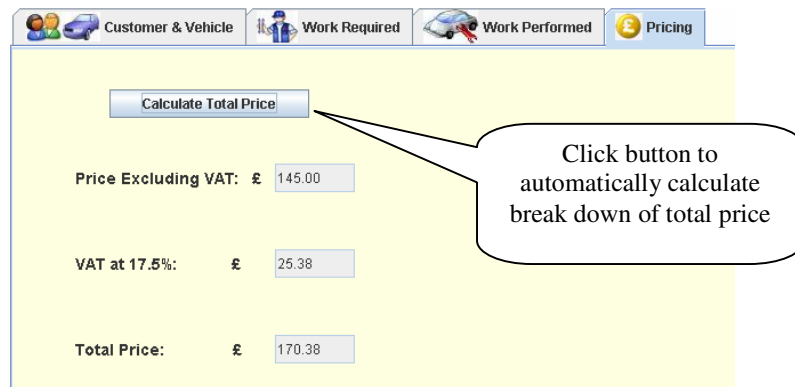
Work Carried Out & Parts Fitted:	Price (£):
Regular service carried out	80.00
Fitted new rear brake pads	40.00
Replaced rear brake light	10.00
Applied new bolt and oil to door hinge	15.00

* Required Field

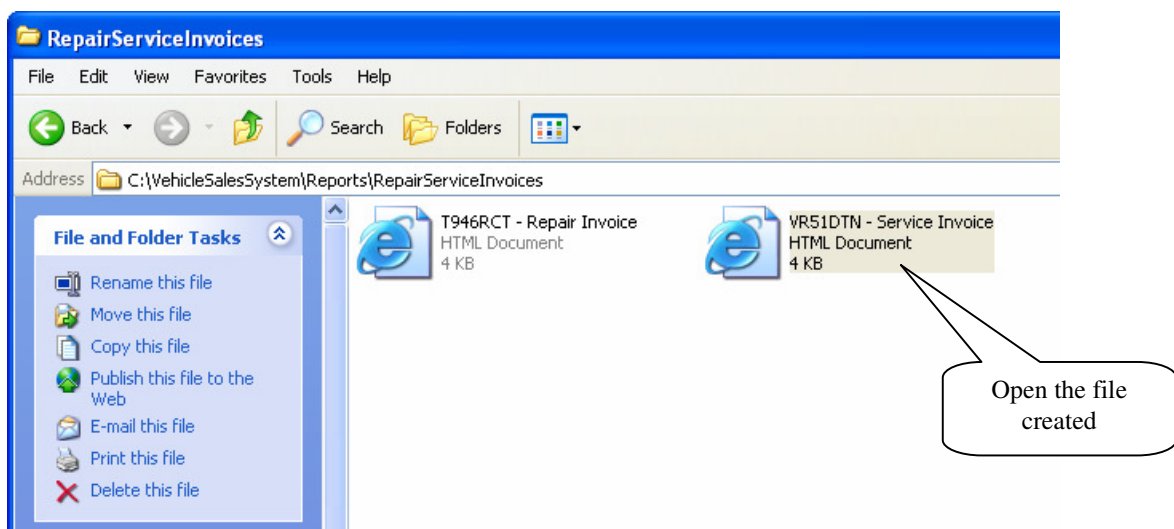
1) Enter the details about work carried out and/or parts fitted in these fields

2) Enter the prices of each work carried out in these fields

Next, go to the Pricing tab and click the Calculate Total Price button. The total price of the work carried out will now be calculated automatically, and a break down of the total price figures will be displayed in the fields. Finally click the *Update Appointment* button at the bottom of the screen. The information about the vehicle scheduled will now be updated.

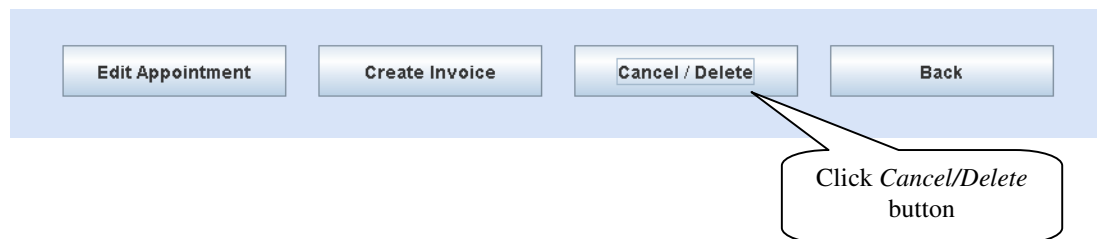


To print out an invoice of the vehicle that was scheduled for a repair or service with work carried out on it, click the *Create Invoice* button on the screen to view the appointment's details. The details will now be written to a HTML file ready to print off. To print the file, navigate to the following directory *C:\VehicleSalesSystem\Reports\RepairServiceInvoices* where the file is created, locate the filename which will include the vehicle's registration number, and open the file. Finally print the file as usually done.



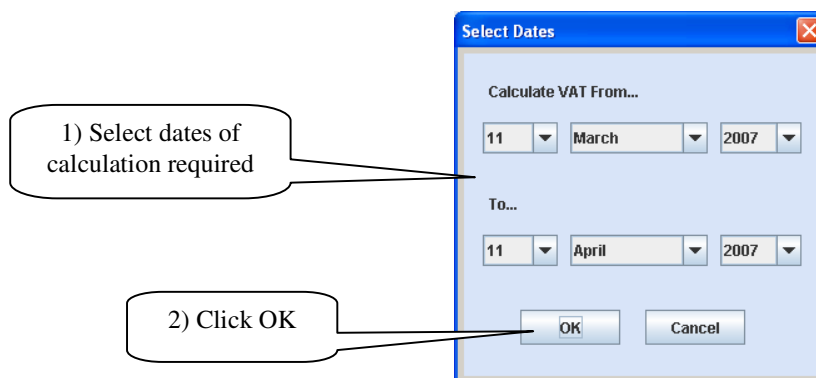
CANCEL / DELETE APPOINTMENT

To cancel or delete a vehicle that has been scheduled for a repair or service, click the *Cancel/Delete* button on the screen to view an appointment's full details. When the message box appears to confirm the cancellation or deletion, click *Yes*. The appointment will now be deleted.



CALCULATE VAT DUE

To calculate the total VAT due on vehicles sold within a specified time period, click the *Calculate VAT* button on the main menu to go to the VAT Calculation screen. When the dates selection box appears, select the dates of VAT calculation required from the drop down boxes and click *OK*.



The details of the sales within the time period specified will now be displayed in the table, along with their individual VAT figures, and the total VAT due to be paid for that period.

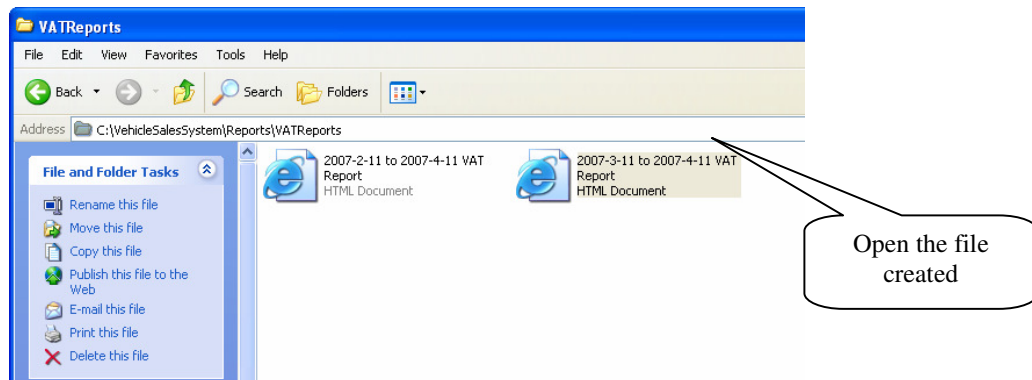
VAT CALCULATION								
2007-3-11 to 2007-4-11								
Reg Num	Vehicle	Supplier	Customer	Date Sold	Price Bought (£)	Price Sold (£)	Gross Profit (£)	VAT Due (£)
YK05RST	Volkswagen Golf	Leeds Grand C...	Mr. Thomas Phi...	2007-04-11	9250	9800	550	96.25
W845PRW	Honda Civic	Fartown Car Sa...	Mr. Andrew Joh...	2007-04-03	3500	4000	500	87.5
NR52YBR	Ford Focus	Fartown Car Sa...	Miss. Kate Willi...	2007-04-08	3500	4000	500	87.5
MV04FDR	Citroen C3	Fartown Car Sa...	Mr. James Sim...	2007-04-01	4500	5000	500	87.5
W726CRX	Vauxhall Astra	Miss. Kate Willi...	Mr. George Fish...	2007-04-11	2000	2200	200	35
AR02HGF	Volkswagen Pa...	Leeds Grand C...	Mr. Graeme Fra...	2007-04-11	5500	6000	500	87.5
CE02NTR	Peugeot 307	Fartown Car Sa...	Mrs. Kelly Ram...	2007-03-20	4500	4750	250	43.75
Y802BTG	Vauxhall Corsa	Fartown Car Sa...	Mr. Jack Porter	2007-03-18	3000	3500	500	87.5
BT03TRE	Toyota Avensis	Fartown Car Sa...	Mr. Alan Redman	2007-04-10	6500	7000	500	87.5
S385REV	BMW 3 Series	Mrs. Tim Foster	Mr. Steven Smith	2007-04-11	6000	7000	1000	175

VAT of each vehicle sold

Total VAT due figure

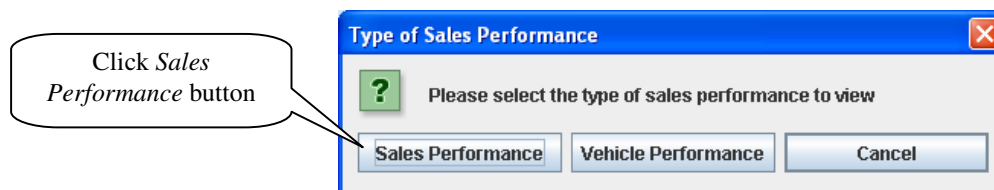
Total Profit (£)	Total VAT Due (£)
5000.0	875.0

To print out the details of the total VAT due calculation, click the *Create Report* button on the screen. This will now create a HTML file with the VAT due information written to the file in order for it to be printed off. To print the file, navigate to the following directory *C:\VehicleSalesSystem\Reports\VATReports* where the file is created, locate the filename which will include the dates of calculation performed, and open the file. The file can now be printed out in the usual way.

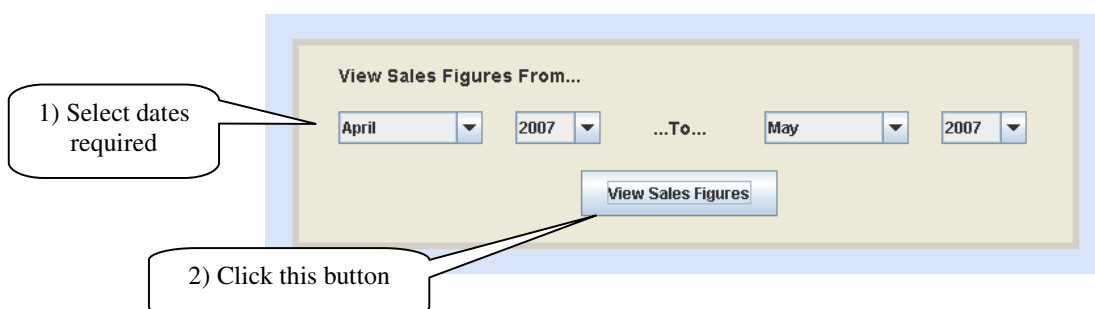


VIEW SALES PERFORMANCE OF BUSINESS

To view the sales performance of the business, click the *Sales Performance* button on the main menu to display the window to select what type of sales performance is required to be viewed. On this window click the *Sales Performance* button.



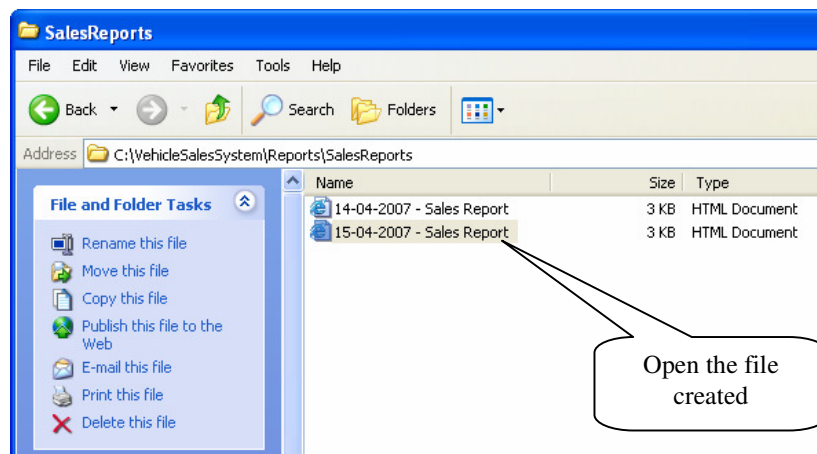
The Sales Performance screen will now be displayed. From the dates selection area, select the periods where sales performance of vehicles is to be viewed, and then click the *View Sales Figures* button.



For each different date periods selected and the *View Sales Figures* button clicked, the sales performance figures for that period will be added to the table.

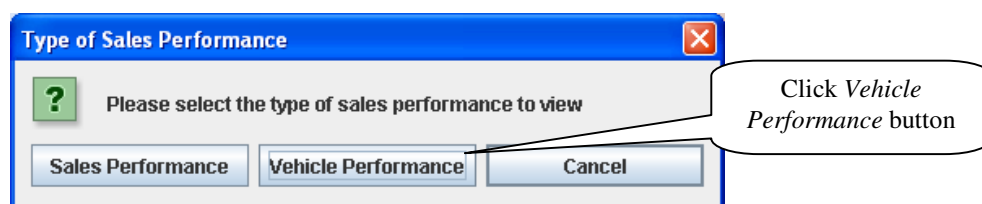
SALES PERFORMANCE							
Date	Total Costs (£)	Purchases	Total Income (£)	No. of Sales	Gross Profit (£)	VAT Paid (£)	Net Profit (£)
Feb 07 - Mar 07	33250.0	10	40750.0	5	7500.0	568.75	6931.25
Mar 07 - Apr 07	66500.0	12	40250.0	6	-26250.0	918.75	-27168.75
Apr 07 - May 07	73250.0	14	45000.0	8	-28250.0	743.75	-28993.75

The details of the sales performance of the business can be printed out, to do this click the *Create Report* button on the screen. This will create a HTML file with the sales performance details in order for it to be printed off. To print the file, navigate to the following directory *C:\VehicleSalesSystem\Reports\SalesReports* where the file is created, locate the filename which will include the date the calculation was carried out, and then open the file. The file can now be printed out in the usual way.

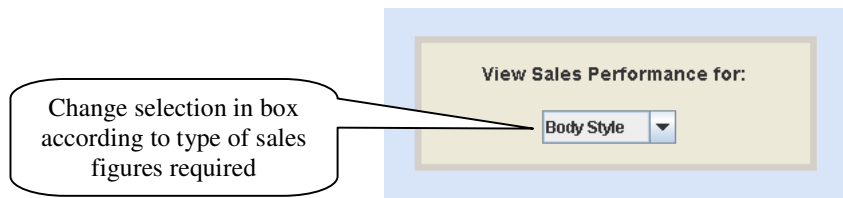


VIEW SALES PERFORMANCE OF VEHICLES

To view the sales performance of vehicles to see which are the most popular and best selling vehicles, click the *Sales Performance* button on the main menu. This will display the box to select the type of sales performance to be viewed, in this box click the *Vehicle Performance* button.



The Vehicle Performance screen will now be displayed, and the sales performance figures initially shown in the table will be about vehicles sold by the make of the vehicle. To view the vehicle sales performance by the body style, select 'Body Style' from the drop down list on the screen.



The sales figures of vehicles in terms of their body styles will now be displayed in the table. To change back to viewing in terms of make, select 'Make' from the box.

VEHICLE PERFORMANCE							
Body Style	Total Stock	Sold	Unsold	Percentage Sold (%)	Avg Cost (£)	Avg Sell Price (£)	Avg Profit (£)
Hatchback	12	9	3	75	6108.0	6727.0	619.0
Coupe	4	2	2	50	6625.0	5375.0	-1250.0
Saloon	9	4	5	44	6250.0	5800.0	-450.0
Estate	5	2	3	40	5840.0	9750.0	3910.0
Convertible	3	1	2	33	7833.0	7000.0	-833.0

CHANGING COMPANY DETAILS

When the system is run for the first time, the company details are set to a default values. It is highly recommended to change the company details according to the company using the system immediately when the system is used for the first time. To change the company details, click the *Administration* button on the main menu to go to the Administration screen. In the Change Company Details section, change the company's details in the fields provided as required, and then click the *Update Company Details* button at the bottom of the page. The company's details will now be changed and can be noticed on the main menu and the printouts.

ADMINISTRATION

Change Company Details

Company Name:

JOHNS CAR SALES

Address 1:

75 Kensington Road

Address 2:

Leeds LS5 3RN

Phone:

0113 295478

Email

info@johncarsales.com

Max Appointments:

5